ITCS447
Lecture 9
IoT Communication and Networking
TCP/IP Model and Access Networks
Standards and Technologies
[RS19]

Asst. Prof. Dr. Thitinan Tantidham

Outline

- Review: History of The Internet & IoT
- Gartner’s Hype Cycle
- IoT Communication and Networking
  - OSI and TCP/IP Networking Models
  - Standards and Technologies
  - Network Equipment
  - IoT Protocol Stack
- IoT Networking Considerations and Challenges
- IoT Connectivity & Access Networks

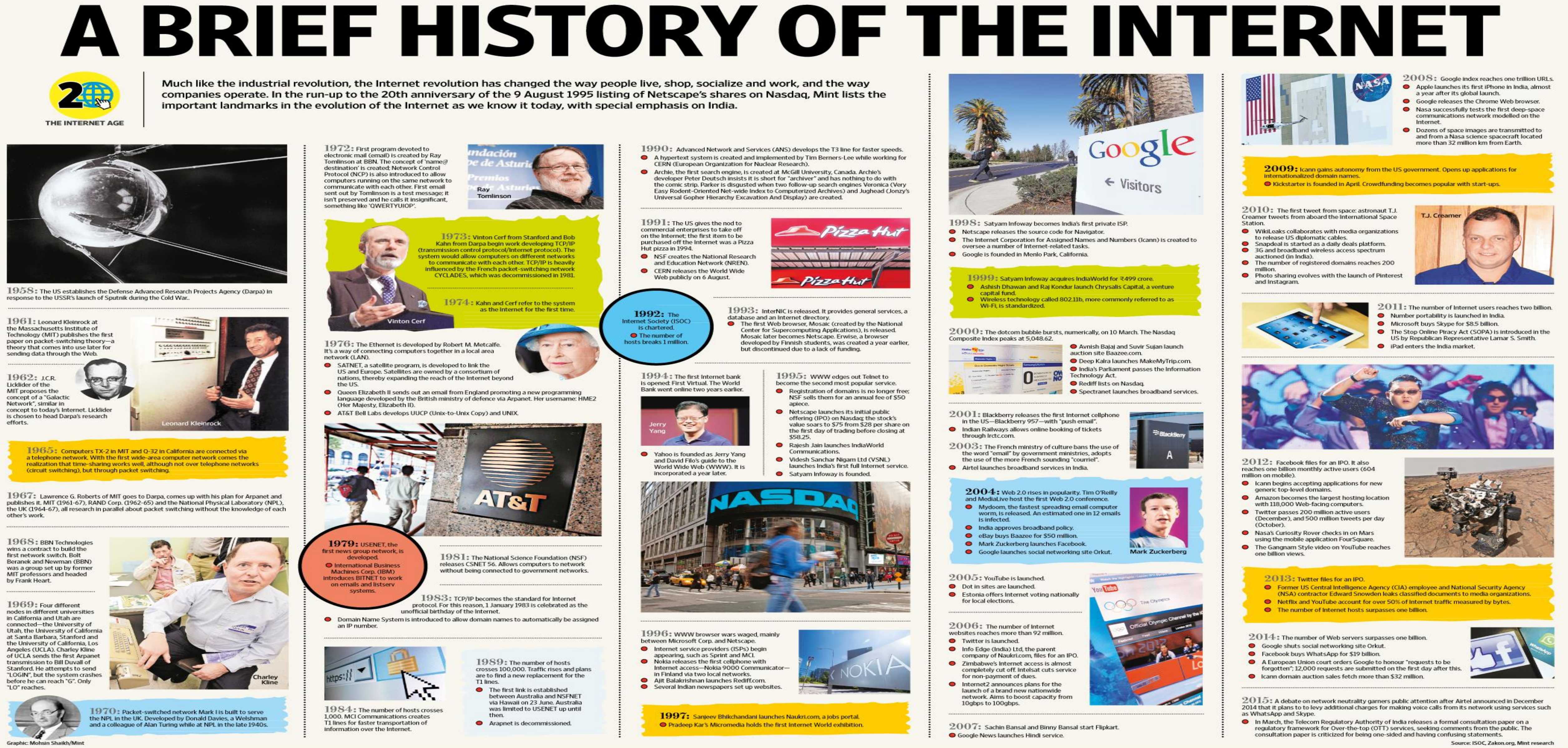
Review: History of the IoT [Lea20]

The following timeline shows the slow progress in connecting things to the Internet.

Read more: https://www.postscapes.com/iot-history/

Year	Device	Reference
1969	The Internet starting with ARPANET (TCP/IP)	
1973	Mario W. Cardullo receives the patent for first RFID tag.	US Patent US 3713148 A
1982	Carnegie Mellon Internet-connected soda machine.	https://www.cs.cmu.edu/~coke/history_long.txt
1989	Internet-connected toaster at Interop'89.	IEEE Consumer Electronics Magazine (Volume: 6, Issue: 1, Jan. 2017)
1991	HP introduces HP LaserJet IIISi: the first Ethernet-connected network printer.	http://hpmuseum.net/display_item.php?hw=350
1993	Internet-connected coffee pot at University of Cambridge (the first Internet-connected camera).	https://www.cl.cam.ac.uk/coffee/qsf/coffee.html
1996	General Motors OnStar (2001 remote diagnostics).	https://en.wikipedia.org/wiki/OnStar
1998	Bluetooth Special Interest Group (SIG) formed.	https://www.bluetooth.com/about-us/our-history
1999	LG Internet Digital DIOS refrigerator.	https://www.telecompaper.com/news/lg-unveils-internetreadyrefrigerator--221266
2000	First instances of the Cooltown concept of pervasive computing everywhere: HP Labs, a system of computing and communication technologies that, combined, create a web-connected experience for people, places, and objects.	https://www.youtube.com/watch?v=U2AkkuIVV-I
2001	First Bluetooth product launched: KDDI Bluetooth-enabled mobile phone.	http://edition.cnn.com/2001/BUSINESS/asia/04/17/tokyo.kddibluetooth/index.html
2005	United Nation's International Telecommunications Union report predicting the rise of IoT for the first time.	http://www.itu.int/osg/spu/publications/internetofthings/internetofThings_summary.pdf
2008	IPSO Alliance formed to promote IP on objects, first IoT-focused alliance.	https://www.ipso-alliance.org
2010	The concept of Smart Lighting formed after success in developing solid-state LED light bulbs.	https://www.bu.edu/smartlighting/files/2010/01/BobK.pdf
2014	Apple creates iBeacon protocol for beacons.	https://support.apple.com/enus/HT202880

Review: History of the Internet



A BRIEF HISTORY OF THE INTERNET

20 THE INTERNET AGE

Much like the industrial revolution, the Internet revolution has changed the way people live, shop, socialize and work, and the way companies operate. In the run-up to the 20th anniversary of the 9 August 1995 listing of Netscape's shares on Nasdaq, Mint lists the important landmarks in the evolution of the Internet as we know it today, with special emphasis on India.

1958: The US establishes the Defense Advanced Research Projects Agency (Darpa) in response to the USSR's launch of Sputnik during the Cold War.

1961: Leonard Kleinrock at the Massachusetts Institute of Technology (MIT) publishes the first paper on packet-switching theory—a theory that comes into use later for sending data through the Web.

1962: J.C.R. Licklider of the MIT proposes the concept of a "Galactic Network", similar in concept to today's Internet. Licklider is chosen to head Darpa's research efforts.

Leonard Kleinrock

1965: Computers TX-2 in MIT and Q-32 in California are connected via a telephone network. With the first wide-area computer network comes the realization that time-sharing works well, although not over telephone networks (circuit switching), but through packet switching.

1967: Lawrence G. Roberts of MIT goes to Darpa, comes up with his plan for Arpanet and publishes it. MIT (1961-67), RAND Corp. (1962-65) and the National Physical Laboratory (NPL), the UK (1964-67), all research in parallel about packet switching without the knowledge of each other's work.

1968: BBN Technologies wins a contract to build the first network switch. Bolt Beranek and Newman (BBN) was a group set up by former MIT professors and headed by Frank Heart.

1969: Four different nodes in different universities in California and Utah are connected—the University of Utah, the University of California at Santa Barbara, Stanford and the University of California, Los Angeles (UCLA). Charley Kline of UCLA sends the first Arpanet transmission to Bill Duvall of Stanford. He attempts to send "LOGIN", but the system crashes before he can reach "G". Only "LO" reaches.

Charley Kline

1970: Packet-switched network Mark I is built to serve the NPL in the UK. Developed by Donald Davies, a Welshman and a colleague of Alan Turing while at NPL in the late 1940s.

Graphic: Mohsin Shaikh/Mint

1972: First program devoted to electronic mail (email) is created by Ray Tomlinson at BBN. The concept of 'name@destination' is created; Network Control Protocol (NCP) is also introduced to allow computers running on the same network to communicate with each other. First email sent out by Tomlinson is a test message; it isn't preserved and he calls it insignificant, something like 'QWERTYUIOP'.

Ray Tomlinson

1973: Vinton Cerf from Stanford and Bob Kahn from Darpa begin work developing TCP/IP (transmission control protocol/Internet protocol). The system would allow computers on different networks to communicate with each other. TCP/IP is heavily influenced by the French packet-switching network CYCLADES, which was decommissioned in 1981.

1974: Kahn and Cerf refer to the system as the Internet for the first time.

Vinton Cerf

1976: The Ethernet is developed by Robert M. Metcalfe. It's a way of connecting computers together in a local area network (LAN).
- SATNET, a satellite program, is developed to link the US and Europe. Satellites are owned by a consortium of nations, thereby expanding the reach of the Internet beyond the US.
- Queen Elizabeth II sends out an email from England promoting a new programming language developed by the British ministry of defence via Arpanet. Her username: HME2 (Her Majesty, Elizabeth II).
- AT&T Bell Labs develops UUCP (Unix-to-Unix Copy) and UNIX.

1979: USENET, the first news group network, is developed.
- International Business Machines Corp. (IBM) introduces BITNET to work on emails and listserv systems.

1981: The National Science Foundation (NSF) releases CSNET 56. Allows computers to network without being connected to government networks.

1983: TCP/IP becomes the standard for Internet protocol. For this reason, 1 January 1983 is celebrated as the unofficial birthday of the Internet.
- Domain Name System is introduced to allow domain names to automatically be assigned an IP number.

1984: The number of hosts crosses 1,000. MCI Communications creates T1 lines for faster transportation of information over the Internet.

1989: The number of hosts crosses 100,000. Traffic rises and plans are to find a new replacement for the T1 lines.
- The first link is established between Australia and NSFNET via Hawaii on 23 June. Australia was limited to USENET up until then.
- Arapnet is decommissioned.

1990: Advanced Network and Services (ANS) develops the T3 line for faster speeds.
- A hypertext system is created and implemented by Tim Berners-Lee while working for CERN (European Organization for Nuclear Research).
- Archie, the first search engine, is created at McGill University, Canada. Archie's developer Peter Deutsch insists it is short for "archiver" and has nothing to do with the comic strip. Parker is disgusted when two follow-up search engines Veronica (Very Easy Rodent-Oriented Net-wide Index to Computerized Archives) and Jughead (Jonzy's Universal Gopher Hierarchy Excavation And Display) are created.

1991: The US gives the nod to commercial enterprises to take off on the Internet; the first item to be purchased off the Internet was a Pizza Hut pizza in 1994.
- NSF creates the National Research and Education Network (NREN).
- CERN releases the World Wide Web publicly on 6 August.

1992: The Internet Society (ISOC) is chartered.
- The number of hosts breaks 1 million.

1993: InterNIC is released. It provides general services, a database and an Internet directory.
- The first Web browser, Mosaic (created by the National Center for Supercomputing Applications), is released. Mosaic later becomes Netscape. Erwise, a browser developed by Finnish students, was created a year earlier, but discontinued due to a lack of funding.

1994: The first Internet bank is opened: First Virtual. The World Bank went online two years earlier.

Jerry Yang
- Yahoo is founded as Jerry Yang and David Filo's guide to the World Wide Web (WWW). It is incorporated a year later.

1995: WWW edges out Telnet to become the second most popular service.
- Registration of domains is no longer free; NSF sells them for an annual fee of $50 apiece.
- Netscape launches its initial public offering (IPO) on Nasdaq; the stock's value soars to $75 from $28 per share on the first day of trading before closing at $58.25.
- Rajesh Jain launches IndiaWorld Communications.
- Videsh Sanchar Nigam Ltd (VSNL) launches India's first full Internet service.
- Satyam Infoway is founded.

1996: WWW browser wars waged, mainly between Microsoft Corp. and Netscape.
- Internet service providers (ISPs) begin appearing, such as Sprint and MCI.
- Nokia releases the first cellphone with Internet access—Nokia 9000 Communicator—in Finland via two local networks.
- Ajit Balakrishnan launches Rediff.com.
- Several Indian newspapers set up websites.

1997: Sanjeev Bhikchandani launches Naukri.com, a jobs portal.
- Pradeep Kar's Micromedia holds the first Internet World exhibition.

1998: Satyam Infoway becomes India's first private ISP.
- Netscape releases the source code for Navigator.
- The Internet Corporation for Assigned Names and Numbers (Icann) is created to oversee a number of Internet-related tasks.
- Google is founded in Menlo Park, California.

1999: Satyam Infoway acquires IndiaWorld for ₹499 crore.
- Ashish Dhawan and Raj Kondur launch Chrysalis Capital, a venture capital fund.
- Wireless technology called 802.11b, more commonly referred to as Wi-Fi, is standardized.

2000: The dotcom bubble bursts, numerically, on 10 March. The Nasdaq Composite Index peaks at 5,048.62.
- Avnish Bajaj and Suvir Sujan launch auction site Baazee.com.
- Deep Kalra launches MakeMyTrip.com.
- India's Parliament passes the Information Technology Act.
- Rediff lists on Nasdaq.
- Spectranet launches broadband services.

2001: Blackberry releases the first Internet cellphone in the US—Blackberry 957—with "push email".
- Indian Railways allows online booking of tickets through Irctc.com.

2003: The French ministry of culture bans the use of the word "email" by government ministries, adopts the use of the more French sounding "courriel".
- Airtel launches broadband services in India.

2004: Web 2.0 rises in popularity. Tim O'Reilly and MediaLive host the first Web 2.0 conference.
- Mydoom, the fastest spreading email computer worm, is released. An estimated one in 12 emails is infected.
- India approves broadband policy.
- eBay buys Baazee for $50 million.
- Mark Zuckerberg launches Facebook.
- Google launches social networking site Orkut.

Mark Zuckerberg

2005: YouTube is launched.
- Dot in sites are launched.
- Estonia offers Internet voting nationally for local elections.

2006: The number of Internet websites reaches more than 92 million.
- Twitter is launched.
- Info Edge (India) Ltd, the parent company of Naukri.com, files for an IPO.
- Zimbabwe's Internet access is almost completely cut off. Intelsat cuts service for non-payment of dues.
- Internet2 announces plans for the launch of a brand new nationwide network. Aims to boost capacity from 10gbps to 100gbps.

2007: Sachin Bansal and Binny Bansal start Flipkart.
- Google News launches Hindi service.

2008: Google index reaches one trillion URLs.
- Apple launches its first iPhone in India, almost a year after its global launch.
- Google releases the Chrome Web browser.
- Nasa successfully tests the first deep-space communications network modelled on the Internet.
- Dozens of space images are transmitted to and from a Nasa science spacecraft located more than 32 million km from Earth.

2009: Icann gains autonomy from the US government. Opens up applications for internationalized domain names.
- Kickstarter is founded in April. Crowdfunding becomes popular with start-ups.

2010: The first tweet from space: astronaut T.J. Creamer tweets from aboard the International Space Station.
- WikiLeaks collaborates with media organizations to release US diplomatic cables.
- Snapdeal is started as a daily deals platform.
- 3G and broadband wireless access spectrum auctioned (in India).
- The number of registered domains reaches 200 million.
- Photo sharing evolves with the launch of Pinterest and Instagram.

T.J. Creamer

2011: The number of Internet users reaches two billion.
- Number portability is launched in India.
- Microsoft buys Skype for $8.5 billion.
- The Stop Online Piracy Act (SOPA) is introduced in the US by Republican Representative Lamar S. Smith.
- iPad enters the India market.

2012: Facebook files for an IPO. It also reaches one billion monthly active users (604 million on mobile).
- Icann begins accepting applications for new generic top-level domains.
- Amazon becomes the largest hosting location with 118,000 Web-facing computers.
- Twitter passes 200 million active users (December), and 500 million tweets per day (October).
- Nasa's Curiosity Rover checks in on Mars using the mobile application FourSquare.
- The Gangnam Style video on YouTube reaches one billion views.

2013: Twitter files for an IPO.
- Former US Central Intelligence Agency (CIA) employee and National Security Agency (NSA) contractor Edward Snowden leaks classified documents to media organizations.
- Netflix and YouTube account for over 50% of Internet traffic measured by bytes.
- The number of Internet hosts surpasses one billion.

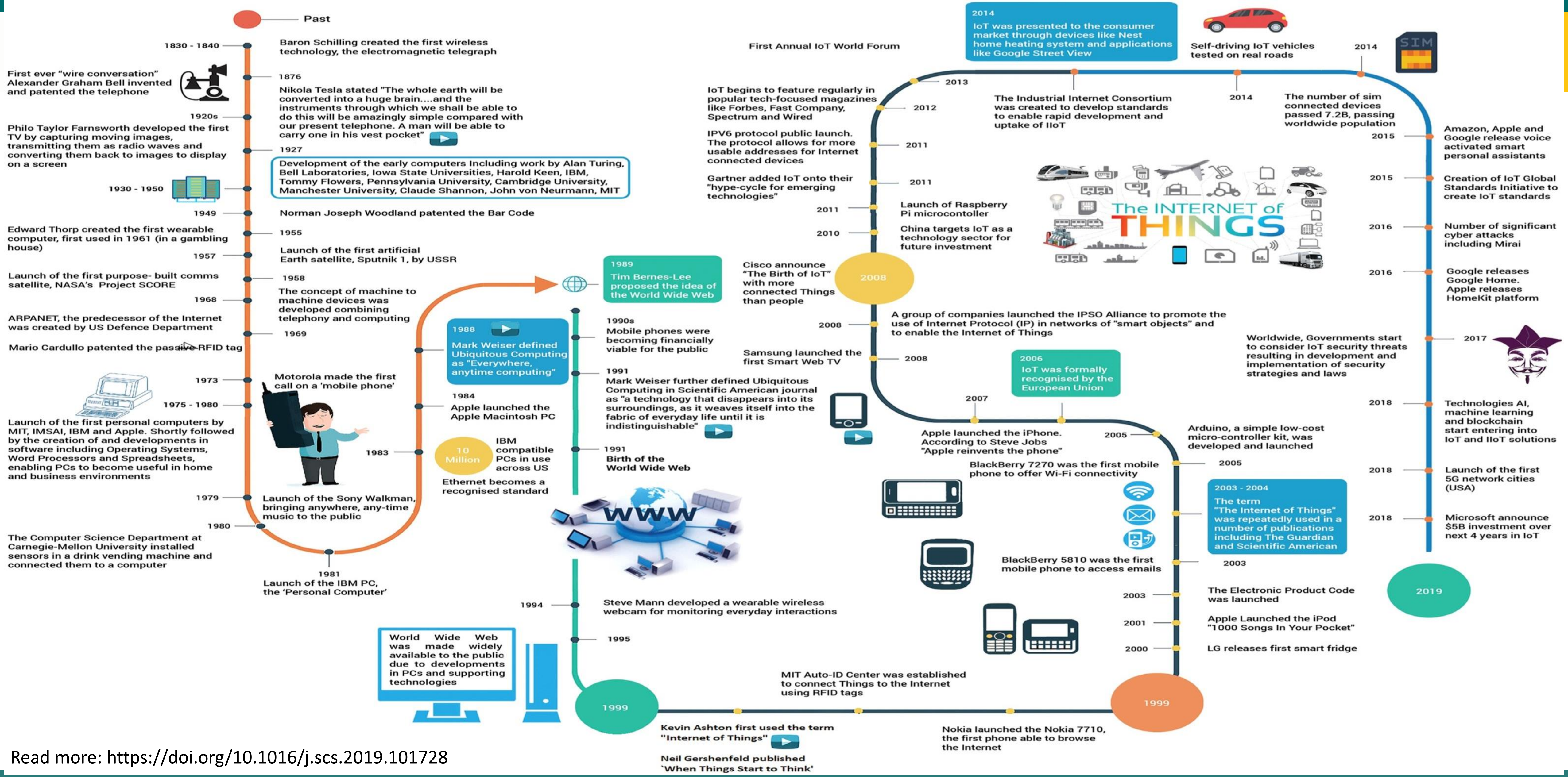
2014: The number of Web servers surpasses one billion.
- Google shuts social networking site Orkut.
- Facebook buys WhatsApp for $19 billion.
- A European Union court orders Google to honour "requests to be forgotten"; 12,000 requests are submitted on the first day after this.
- Icann domain auction sales fetch more than $32 million.

2015: A debate on network neutrality garners public attention after Airtel announced in December 2014 that it plans to to levy additional charges for making voice calls from its network using services such as WhatsApp and Skype.
- In March, the Telecom Regulatory Authority of India releases a formal consultation paper on a regulatory framework for Over-the-top (OTT) services, seeking comments from the public. The consultation paper is criticized for being one-sided and having confusing statements.

Source: ISOC, Zakon.org, Mint research

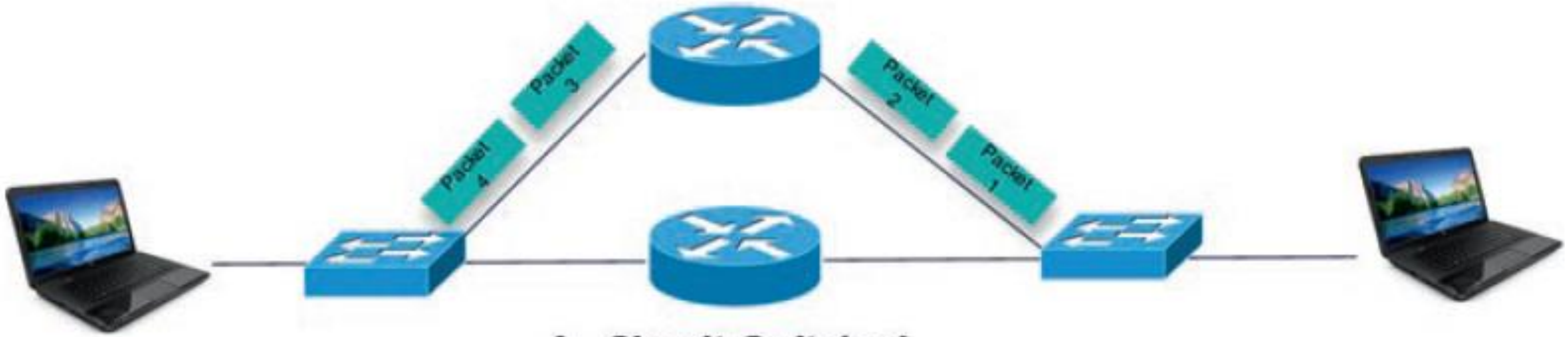
Read more: https://www.livemint.com/Industry/bQkp9arbe4mlaJHQA1xOjO/A-brief-history-of-the-Internet.html

Review: History of the Internet & IoT [AAH+19]

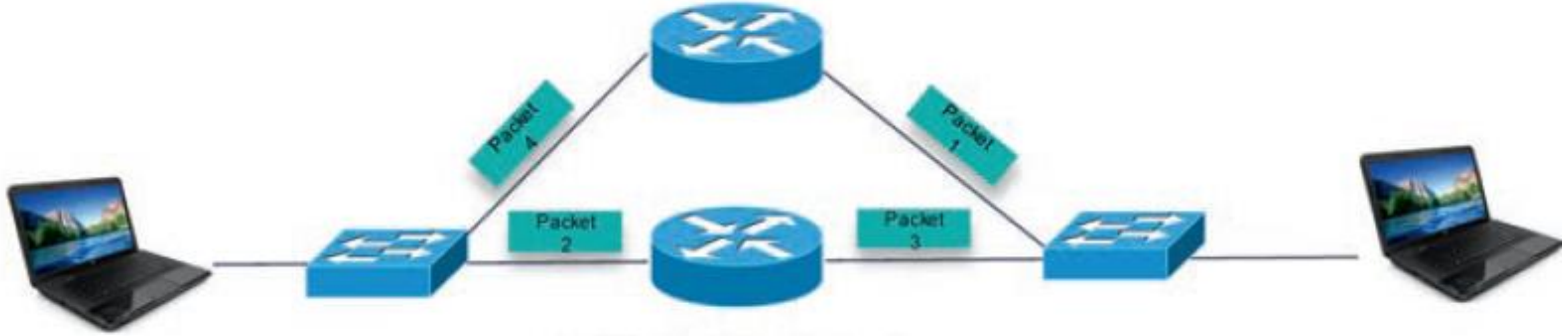


Read more: https://doi.org/10.1016/j.scs.2019.101728

Review: Packet Switching vs Circuit Switching

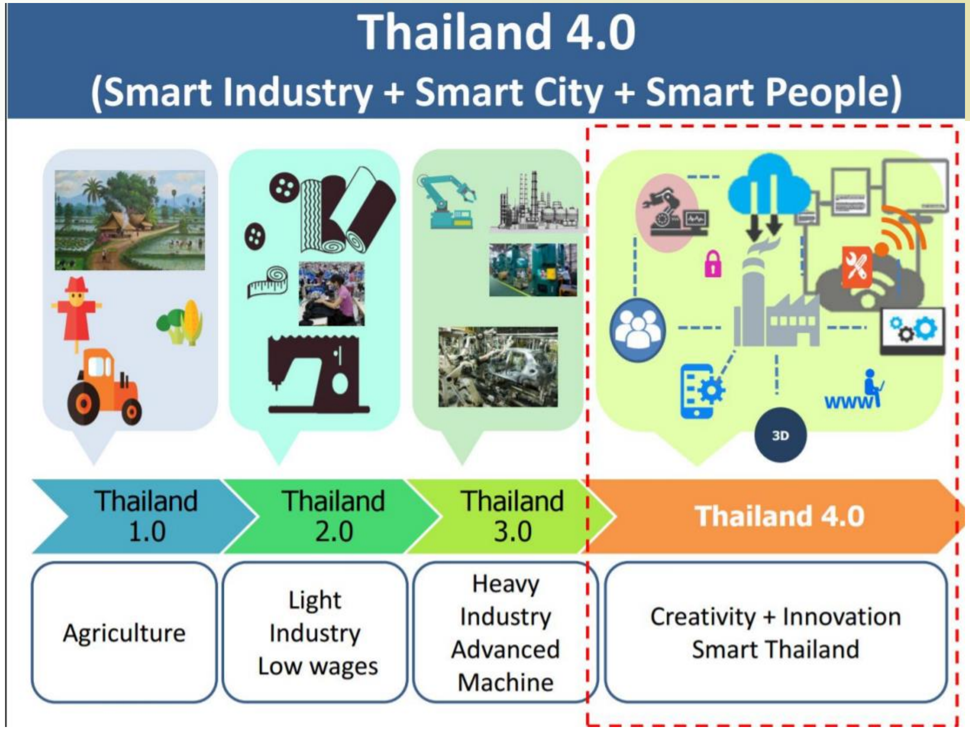


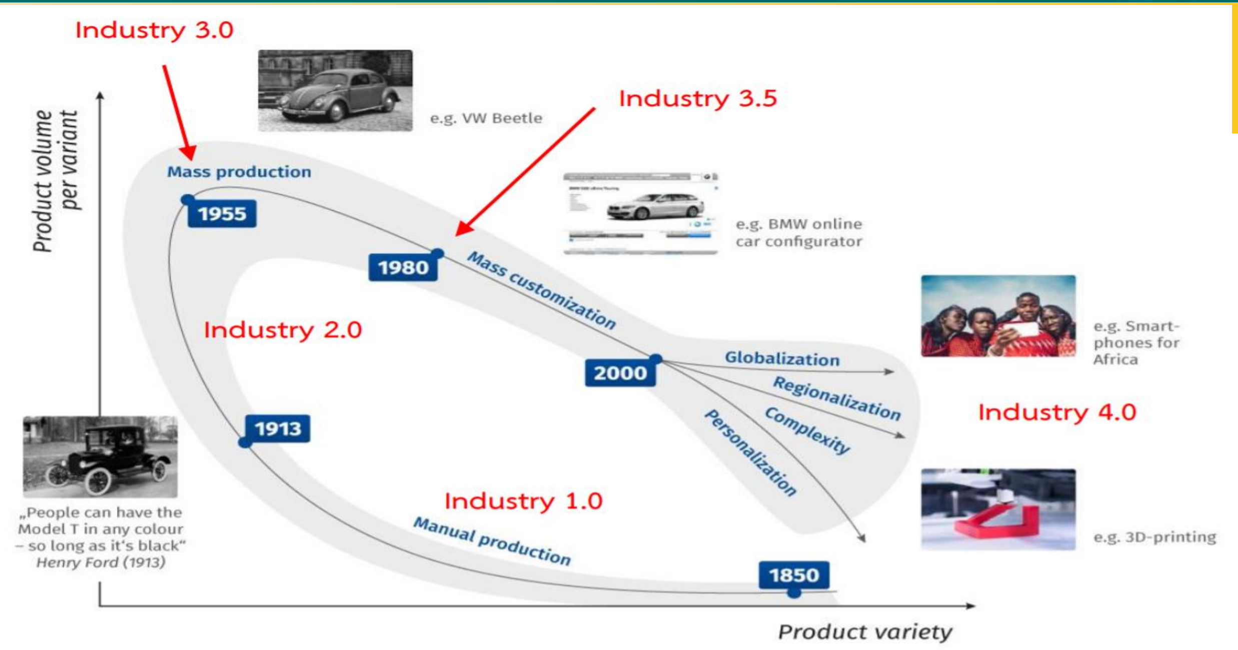
A. Circuit-Switched
Packets Follows Same Path & Arrive in Order

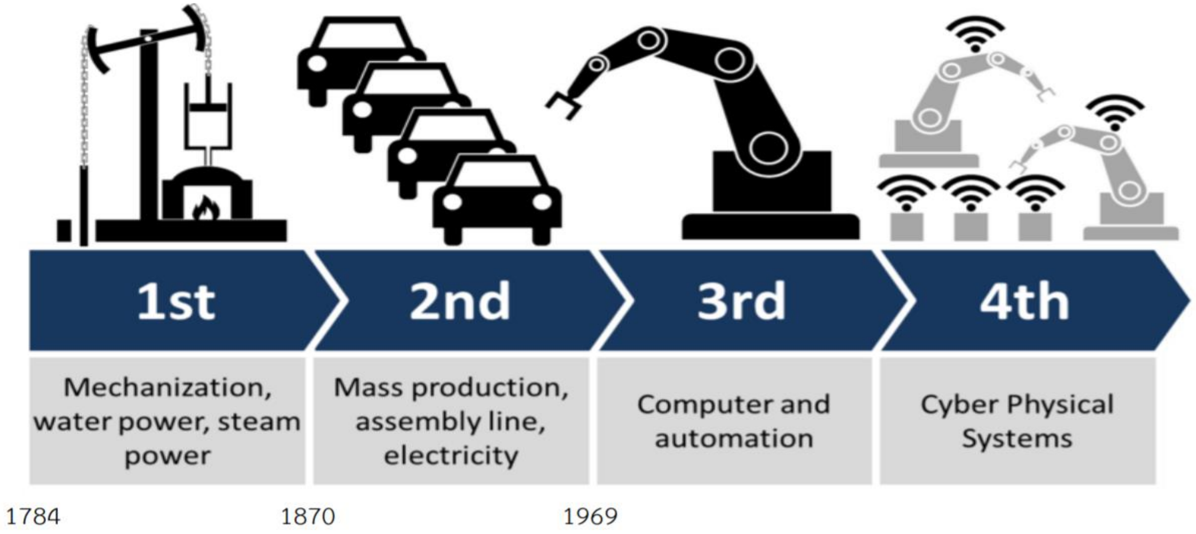


B. Packet-Switched
Packets May Follow Different Paths & Arrive Out-of-Order

Review: Thailand 4.0 vs Industry 4.0



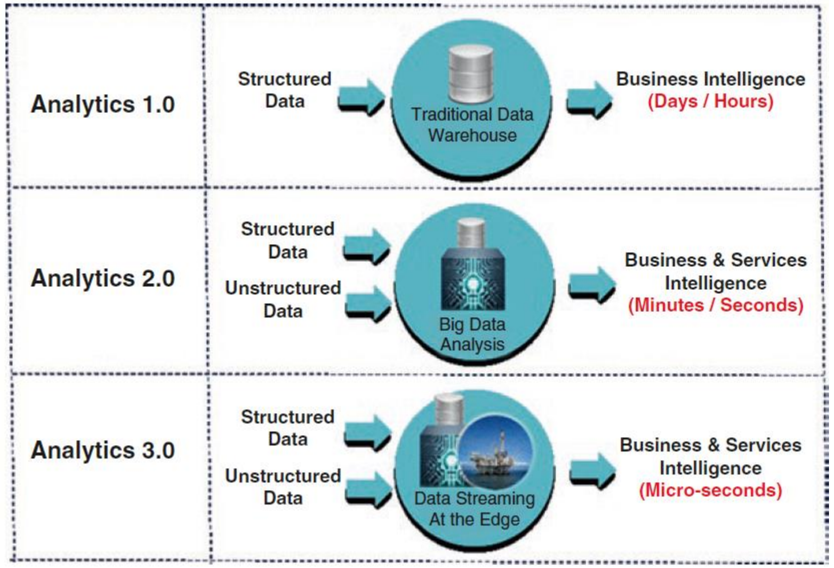




http://www.stta.or.th/images/pdf/Thailand4/Thailand4.pdf

Review: Analytics 1.0/2.0/3.0

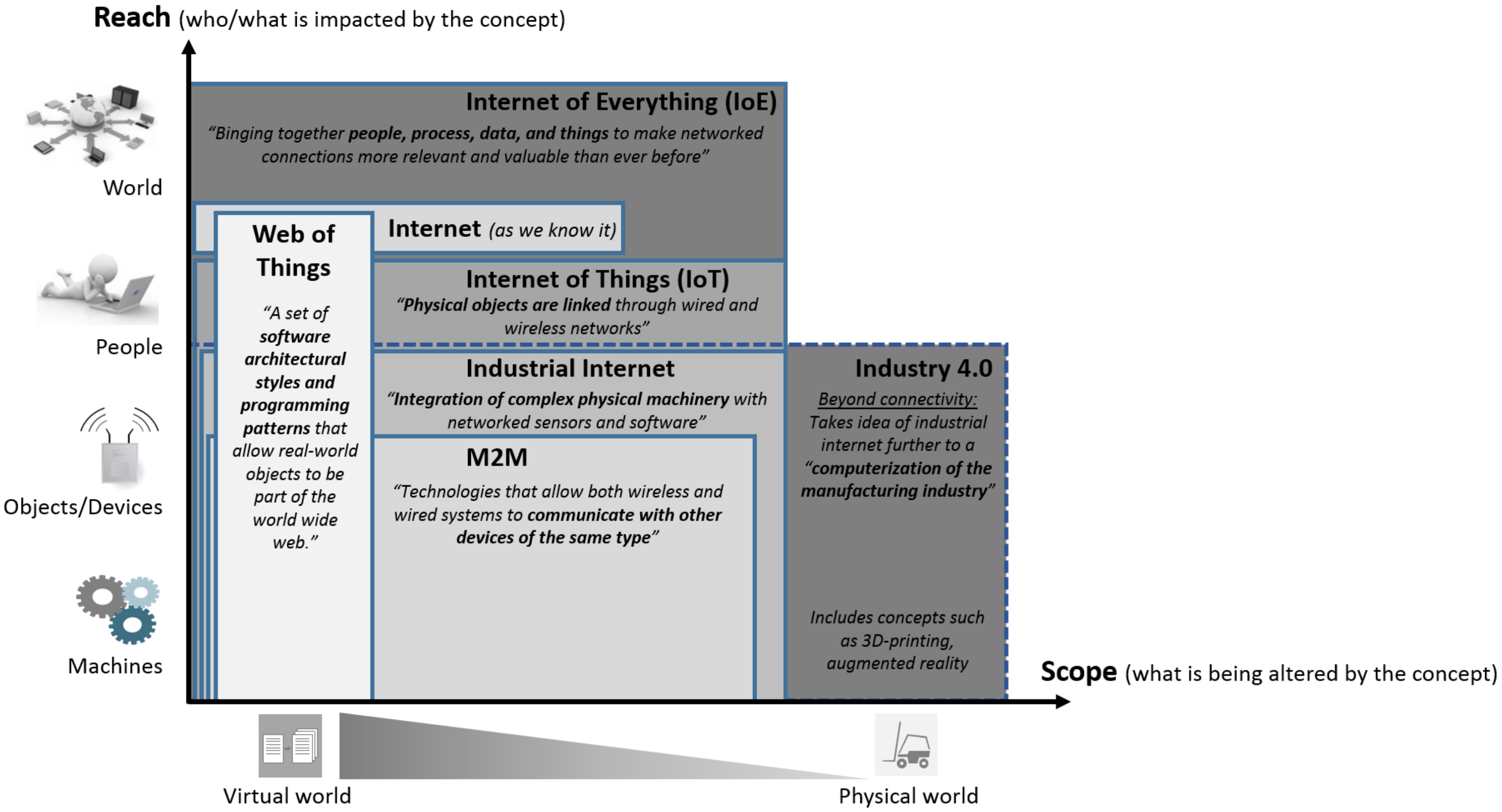
	Analytics 1.0	Analytics 2.0	Analytics 3.0
Collected data type	Structured	Structured and unstructured	Structured and unstructured
Data analysis location	Centralized data center	Centralized data center	At edge and in data center
Time to analyze data	Days–hours	Hours–minutes	Seconds–microseconds
Data volume	Small data	Big data	Big data



Review: M2M vs IoT

M2M	IoT
Machines	Sensors
Hardware-based	Software-based
Vertical applications	Horizontal applications
Deployed in a closed system	Connects to a larger network
Machines communicating with machines	Machines communicating with machines, humans with machines, machines with humans
Uses non-IP protocol	Uses IP protocols
Can use the cloud, but not required to	Uses the cloud
Machines use point-to-point communication, usually embedded in hardware	Devices use IP networks to communicate
Often one-way communication	Back and forth communication
Main purpose is to monitor and control	Multiple applications; multilevel communications
Operates via triggered responses based on an action	Can, but does not have to, operate on triggered responses
Limited integration options, devices must have complementary communication standards	Unlimited integration options, but requires software that manages communications/protocols
Structured data	Structured and unstructured data

Review: M2M vs IoT vs IoE

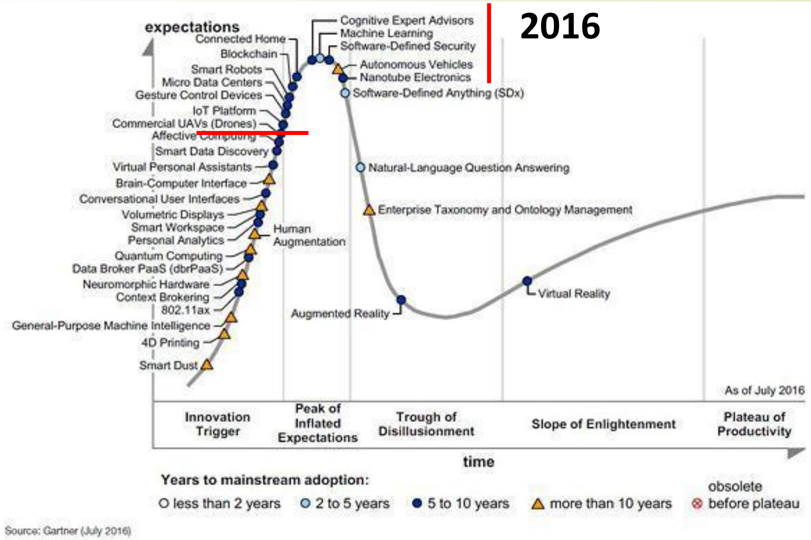


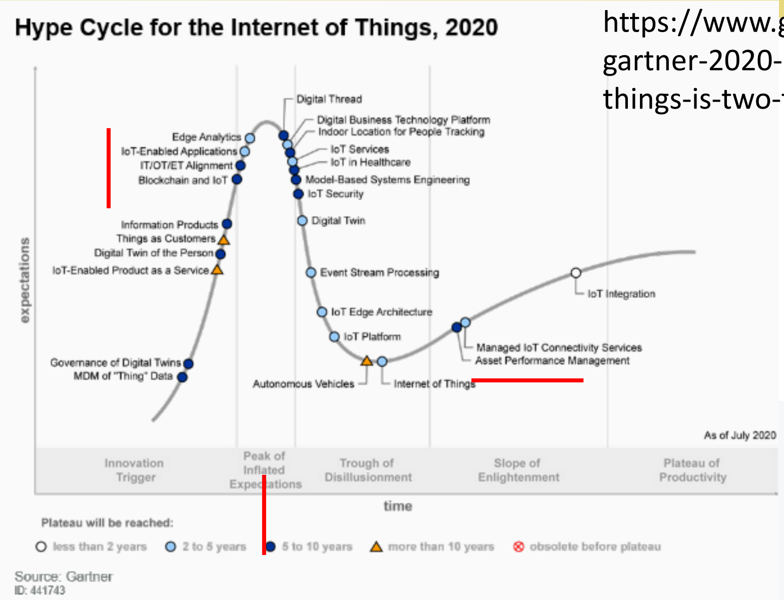
Vikipedia, McKinsey, IoT Analytics

https://iot-analytics.com/internet-of-things-definition/

Gartner's Hype Cycle

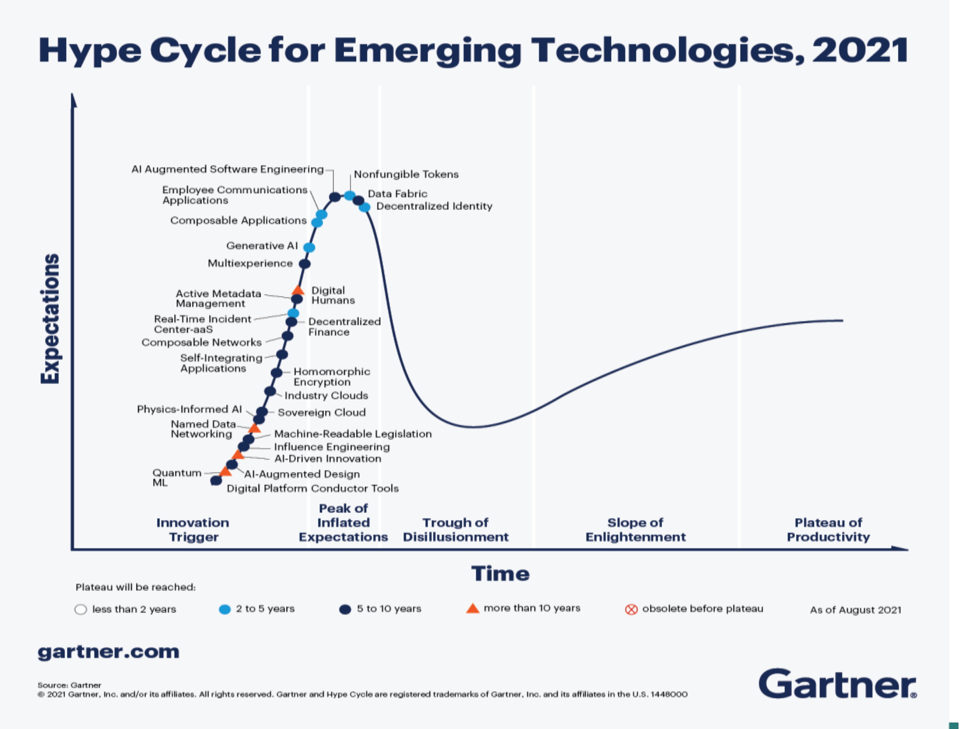
Internet of Things





https://www.gartner.com/en/newsroom/press-releases/2020-09-09-gartner-2020-hype-cycle-for-supply-chain-strategy-shows-internet-of-things-is-two-to-five-years-away-from-transformational-impact

https://www.zdnet.com/article/gartner-releases-its-2021-emerging-tech-hype-cycle-heres-whats-in-and-headed-out/



- Artificial intelligence's impact on generating code, augmenting design and innovation is all 5- to 10-years away.
- Composable is going to be a key buzzword for applications and networks.
- Industry clouds are just beginning on the hype cycle with a plateau reached in 5- to 10-years. That take is interesting given industry clouds are everywhere from multiple vendors.
- Digital humans are being talked about a good bit, but Gartner reckons the technology is more than 10 years away from productivity gains. Quantum-based machine learning is also more than 10 years out.

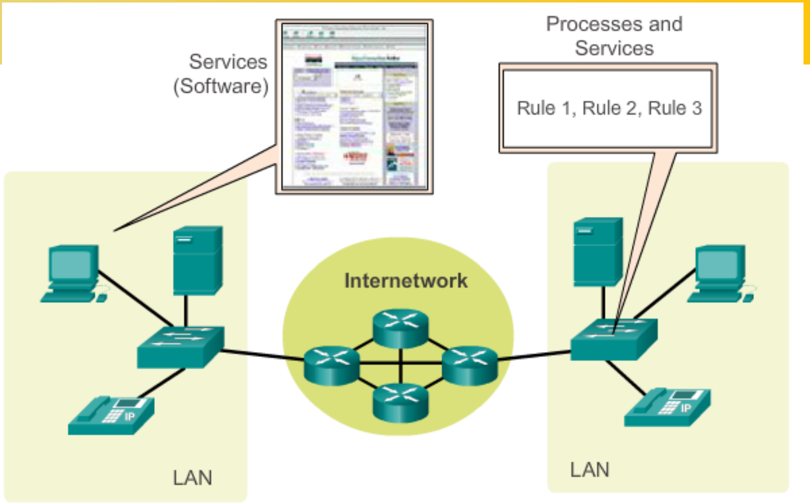
https://en.wikipedia.org/wiki/Gartner_hype_cycle#Five_phases

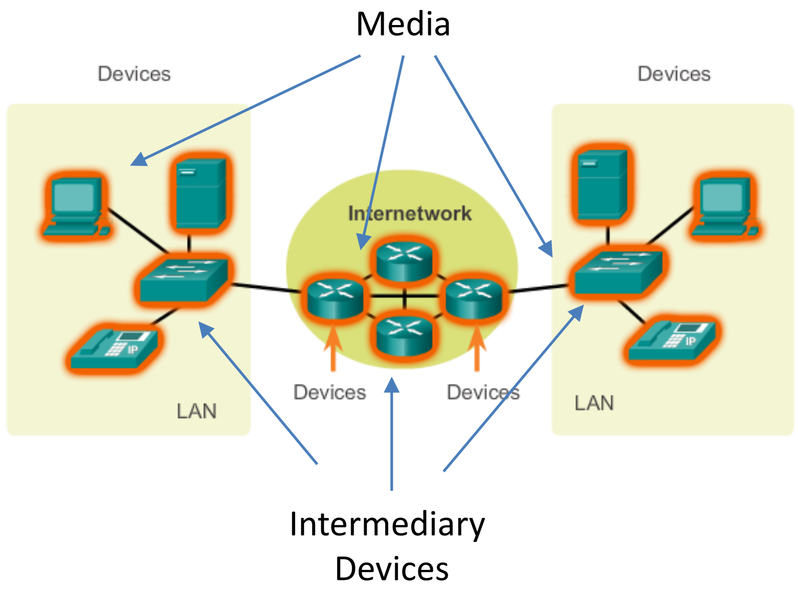
IoT Communication and Networking

- OSI and TCP/IP Networking Models
- Standards and Technologies
- Network Equipment
- IoT Protocol Stack
  - Network Access and Physical Layer
  - Internet Layer
  - Application Layer

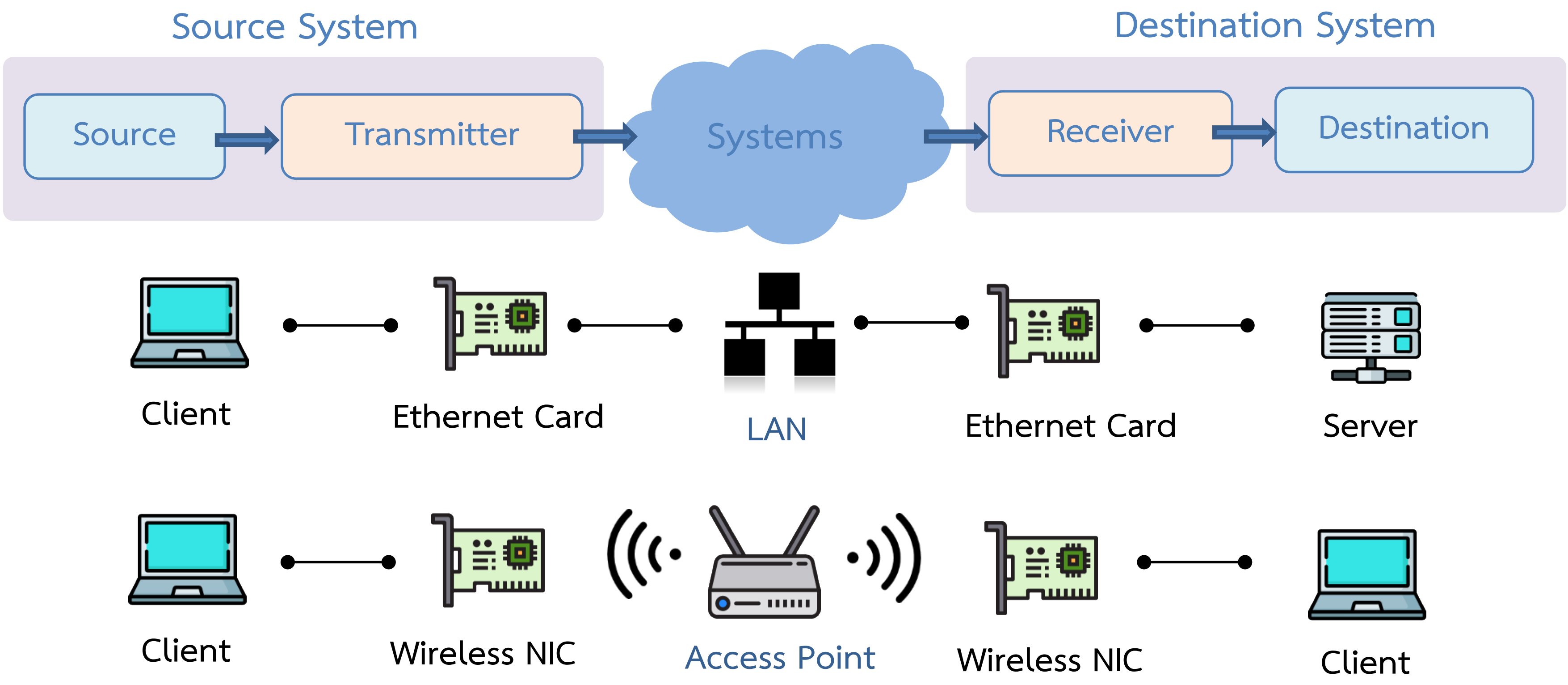
Network Components (1/2)

- (End) Devices or Hosts: Sender/Receiver or Source/Destination
  - Either the source or destination of a message
  - Hosts: Servers -computers provide information to end devices.
    Clients –computers send requests to the servers to retrieve information.
  - IoT devices (e.g. cameras, door locks, doorbells, refrigerators, audio/visual systems, and various sensors)
- (Intermediary Network) Devices
  - Connect multiple individual networks to form an internetwork
  - Connect the individual end devices to the network
  - Ensure data flows across the network
  - Provide connectivity
- (Network) Media
  - Provide the pathway for data transmission
  - Interconnect devices
  - Wired e.g. copper and fiber optic and Wireless e.g. WiFi and Bluetooth
- Processes and Services
  - Services provide information in response to a request. They include network applications e.g. file, email, and web services.
  - Processes provide the functionality that directs and moves the messages through the network e.g. routing and forwarding operations on a router device.





Network Components (2/2)



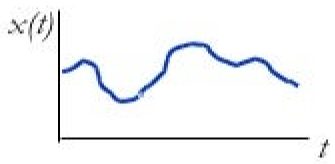
Transmission Media (Wired or Guided)/Wireless or Unguided)

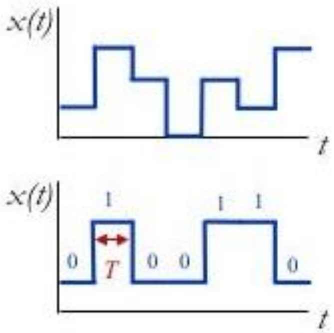
NIC: Network Interface Card

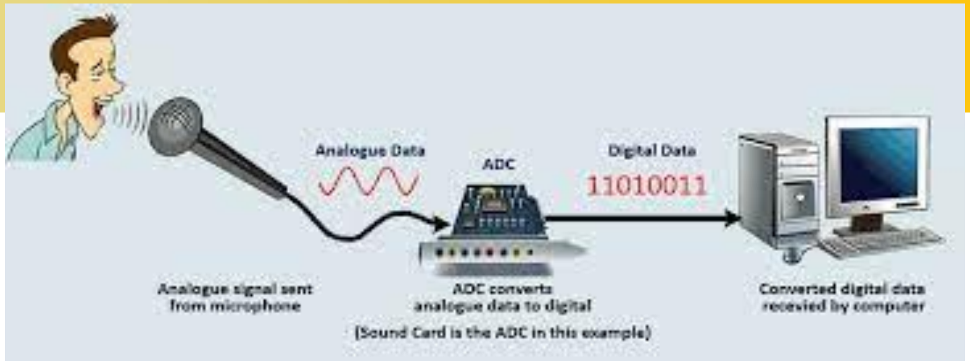
Analog/Digital Data and Signals

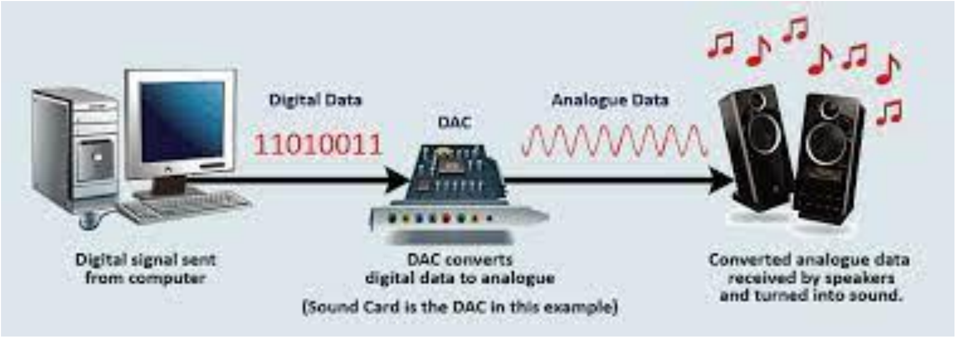
Table 2-1 Four combinations of data and signals

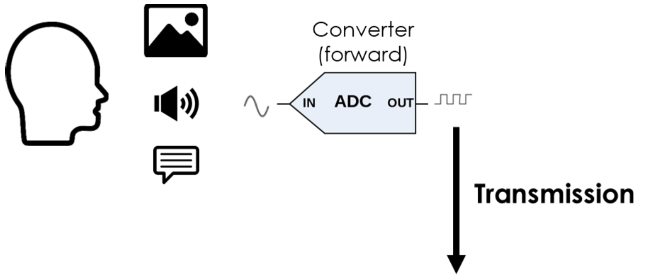
Data	Signal	Encoding or Conversion Technique	Common Devices	Common Systems
Analog	Analog	Amplitude modulation Frequency modulation	Radio tuner TV tuner	Telephone AM and FM radio Broadcast TV Cable TV
Digital	(Square-wave) Digital	NRZ-L NRZI Manchester Differential Manchester Bipolar-AMI 4B/5B	Digital encoder	Local area networks Telephone systems
Digital	(Discrete) Analog	Amplitude shift keying Frequency shift keying Phase shift keying	Modem	Dial-up Internet access DSL Cable modems Digital Broadcast TV
Analog	Digital	Pulse code modulation Delta modulation	Codec	Telephone systems Music systems

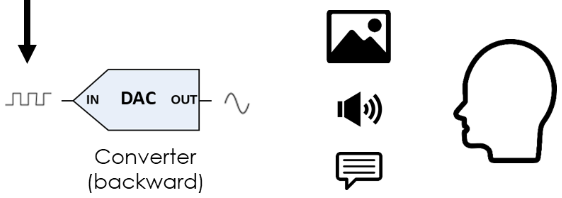












ADC: Analog to Digital Converter
DAC: Digital to Analog Converter

Communication Protocol (1/2)

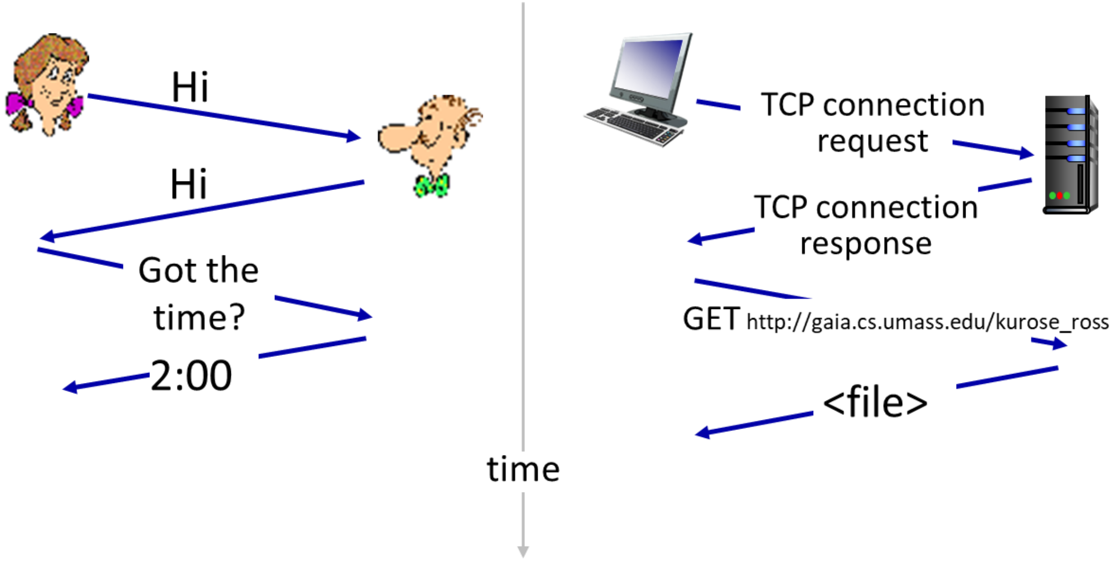
- **A communication protocol** is a **system** of **rules** that **allow two or more entities of a communications system to transmit information** via any kind of variation of a physical quantity.
- The **protocol** defines the **syntax**, **semantics** and **synchronization** of communication and **possible error recovery methods**.
- **Protocols** may be implemented by hardware, software, or a combination of both.

Syntax: rules of communication (e.g. grammar)
Semantics: means of symbols (e.g. vocabulary)
Synchronization: timing of conversation (e.g. timing)

Communication Protocol (2/2)

- A protocol is analogous to “communication language” in human terms.

Computer Communication Protocol	Human Communication Language
Syntax	Grammar
Semantics	Vocabulary
Synchronization	Conversation



OSI vs TCP/IP

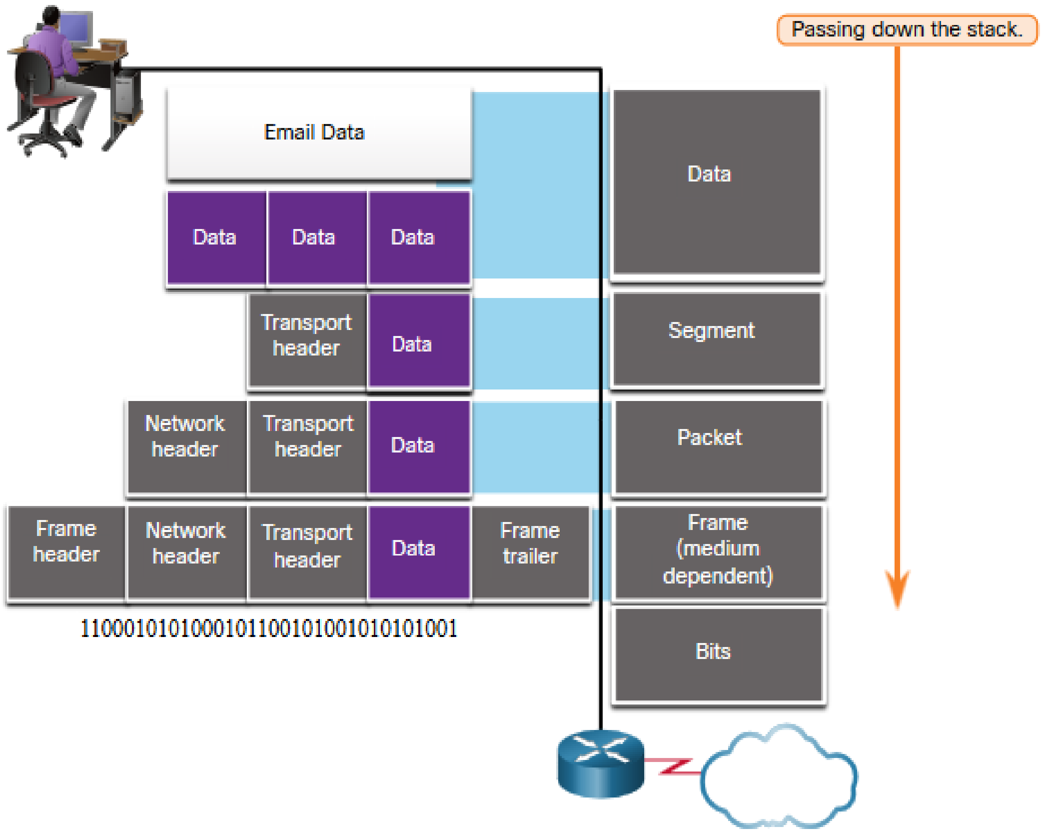
OSI is introduced by ISO (International Organization for Standardization) 1984.

TCP/IP

OSI layer	Main function	Examples of main devices	Examples of main protocol	
Application	Provides network services to the end host's applications	Server, laptops, PCs	HTTPS, FTP, Telent, SSH	
Presentation	Ensures the data can be understood between two end hosts	N/A Application or Network Operating System	Data encoding, data formatting, and serialization	
Session	Manages multiple sessions between end hosts	N/A Application or Network Operating System	Connection management, error recovery	Authentication e.g. login/logout
Transport	Establishes end-to-end connectivity and ensures reliable data delivery	Firewalls	TCP, UDP	
Network	Connectivity and path selection based on logical addresses	Routers, firewalls	IPv4, IPv6	
Data link	Defines data format for transmission	Switches, APs	IEEE 802.1 (Ethernet), PPP	
Physical	Defines physical media access and properties	Fiber optics, category 5 cables, coaxial cables	IEEE 802.3	

OSI (Open Systems Interconnection)
https://en.wikipedia.org/wiki/OSI_model

Data Encapsulation/Decapsulation

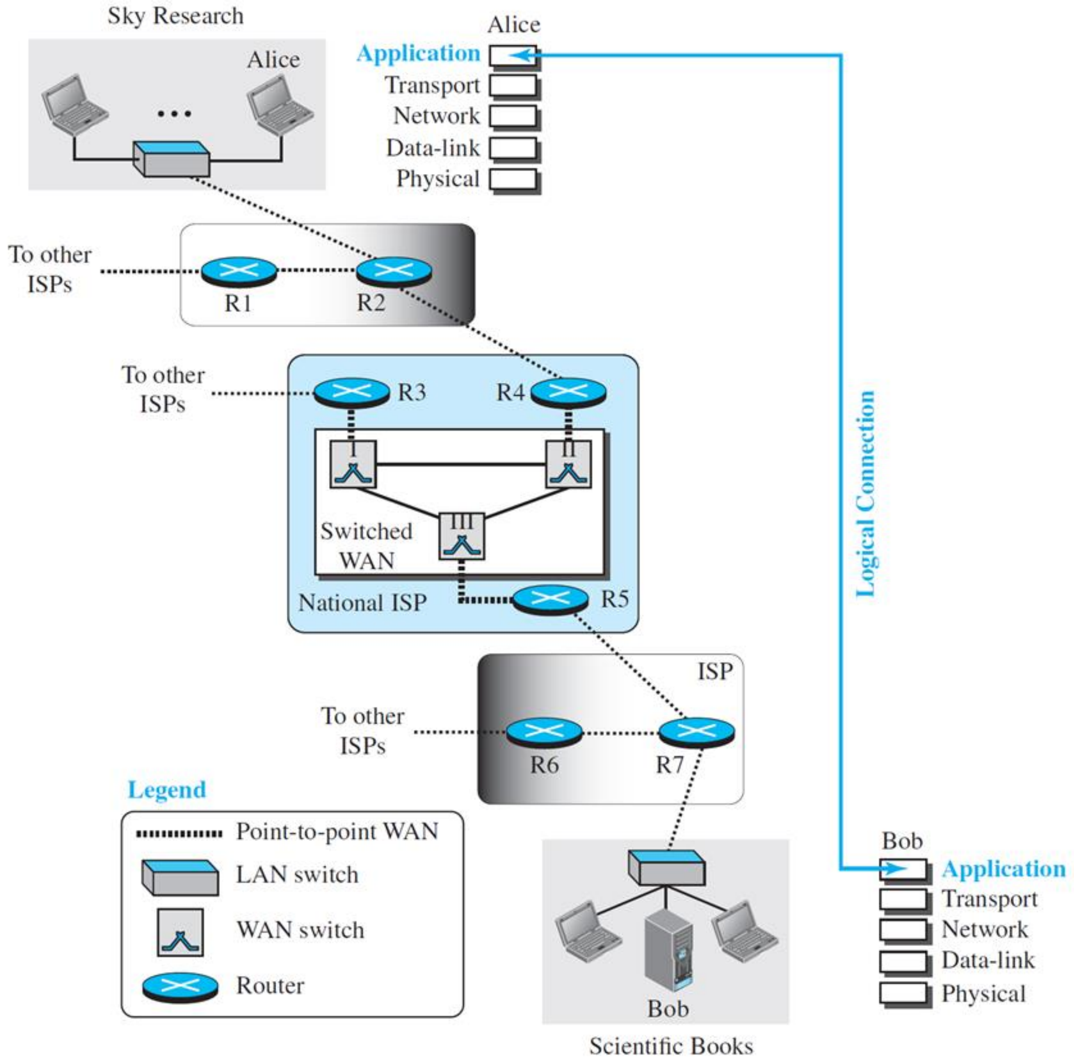


Encapsulation is the process where protocols add their information to the data.

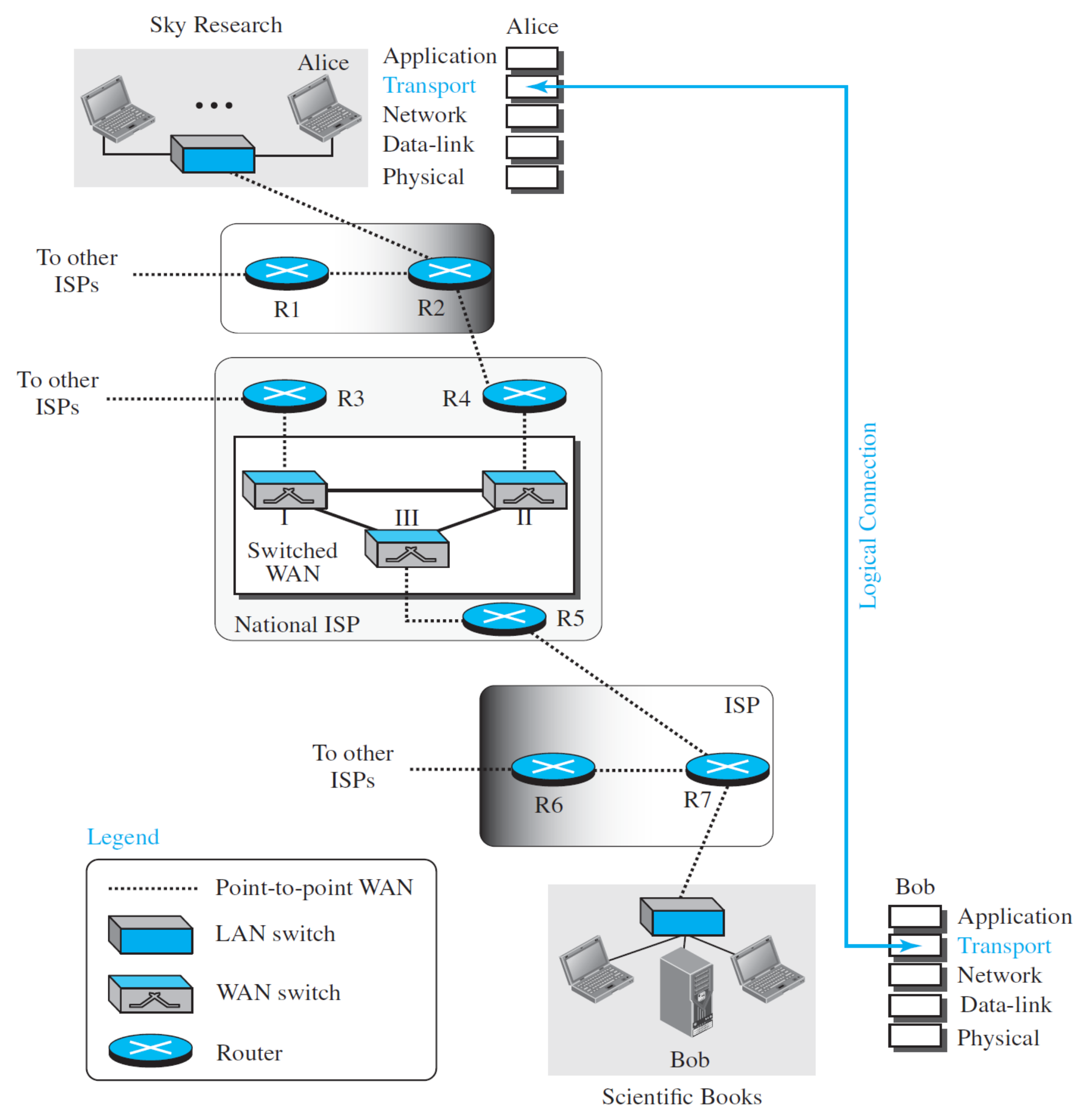
- At each stage of the process, a PDU has a different name to reflect its new functions.
- There is no universal naming convention for PDUs, in this course, the PDUs are named according to the protocols of the TCP/IP suite.
- PDUs passing down the stack are as follows:

1. Data (Data Stream)
2. Segment
3. Packet
4. Frame
5. Bits (Bit Stream)

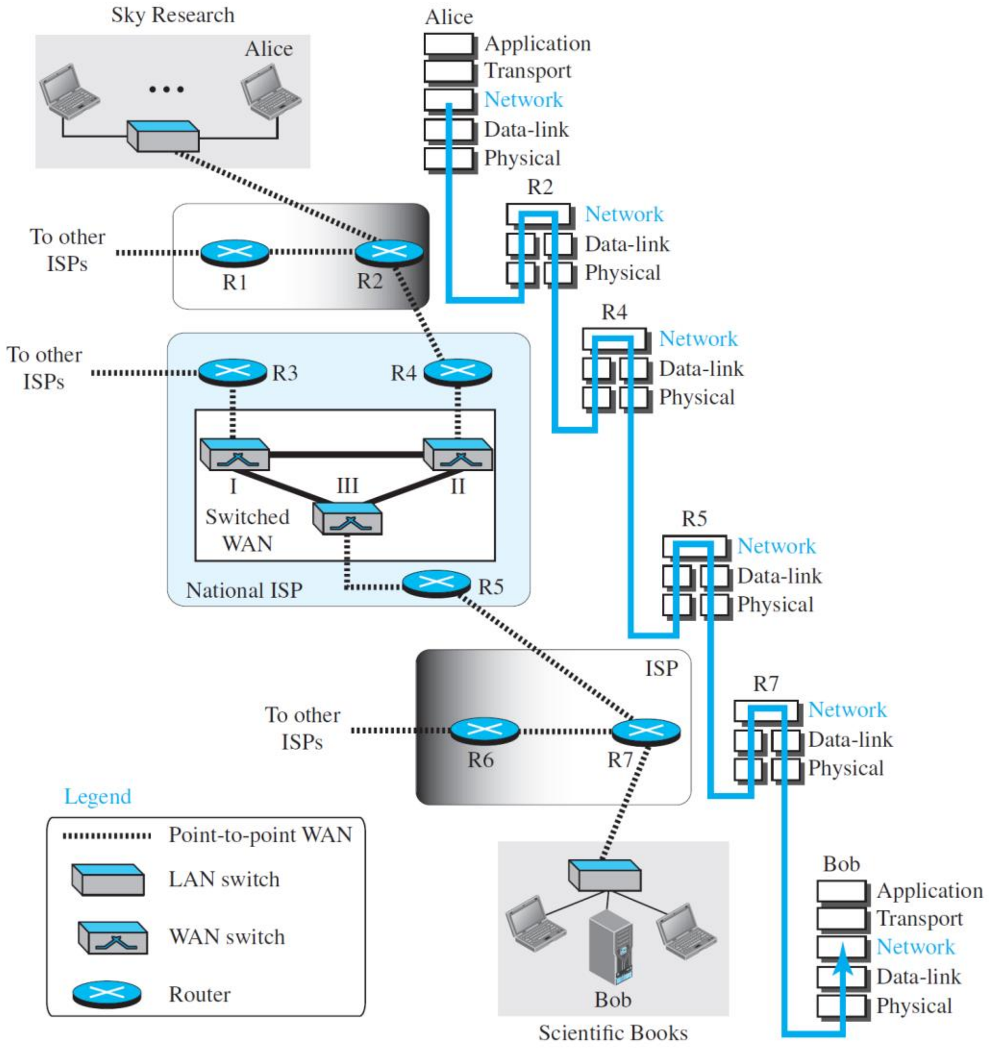
TCP/IP: Application Layer Communication

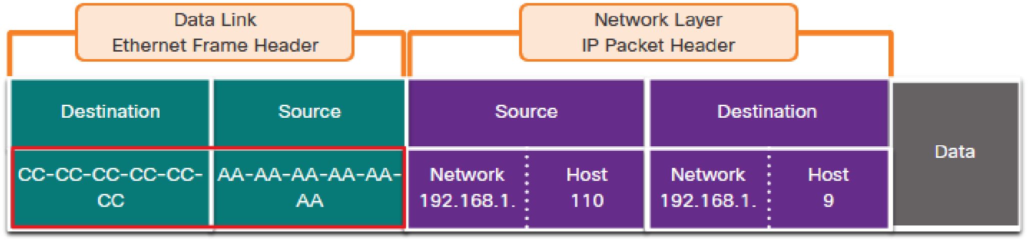


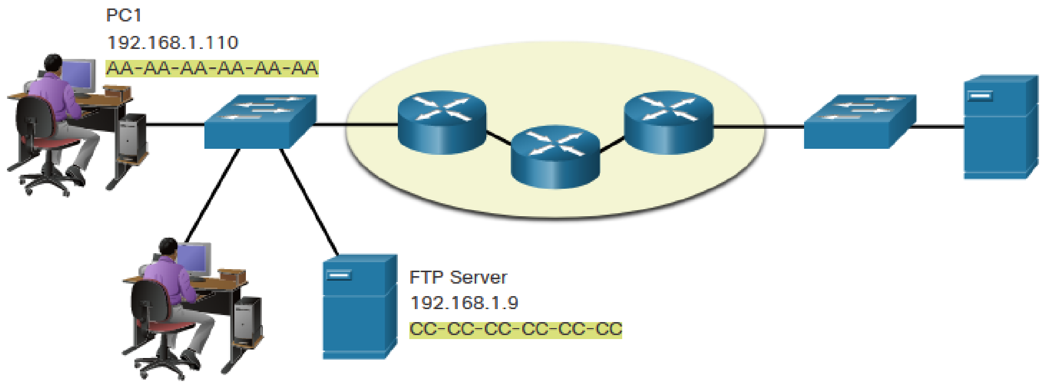
TCP/IP: Transport Layer Communication



TCP/IP: Network Layer Communication





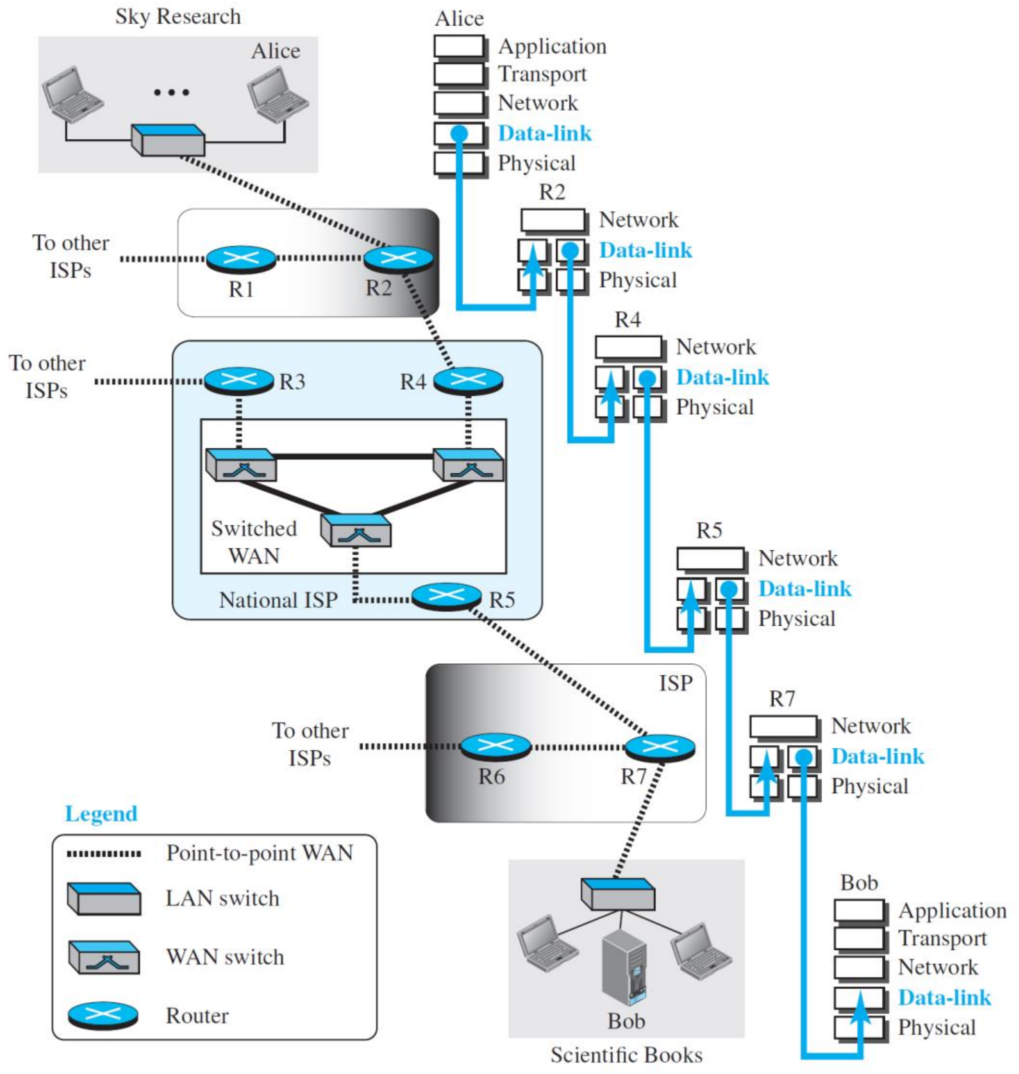


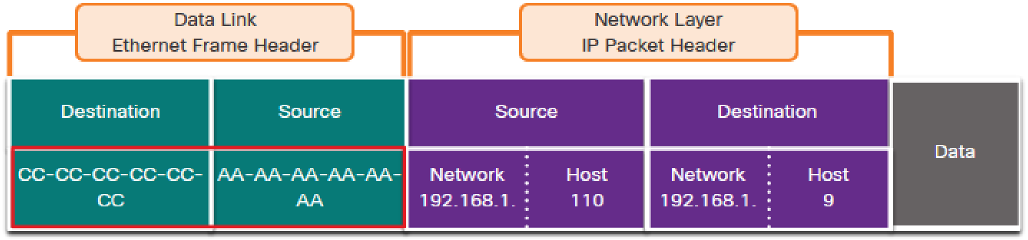
An IP address contains two parts:

- **Network portion (IPv4) or Prefix (IPv6)**
  - The left-most part of the address indicates the network group which the IP address is a member.
  - Each LAN or WAN will have the same network portion.
- **Host portion (IPv4) or Interface ID (IPv6)**
  - The remaining part of the address identifies a specific device within the group.
  - This portion is unique for each device on the network.

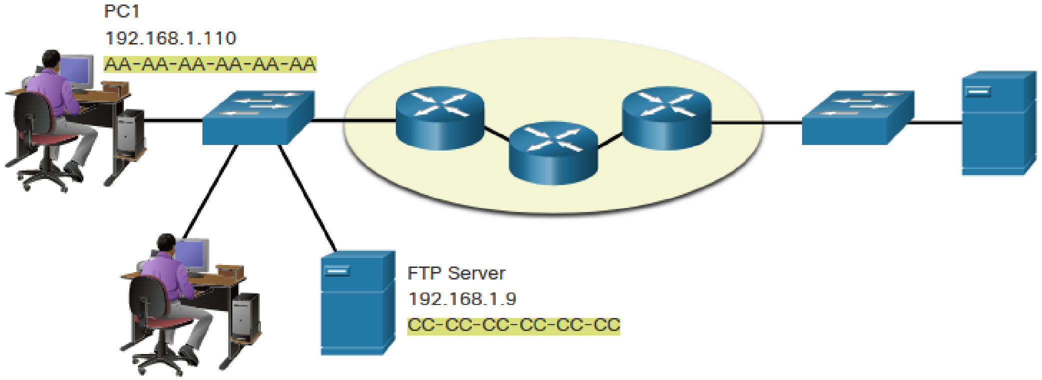
OSI and TCP/IP Networking Models

TCP/IP: Datalink Layer Communication (1/2)





Same IP Network



When devices are on the same Ethernet network the data link frame will use the actual **MAC address** of the **destination NIC**.

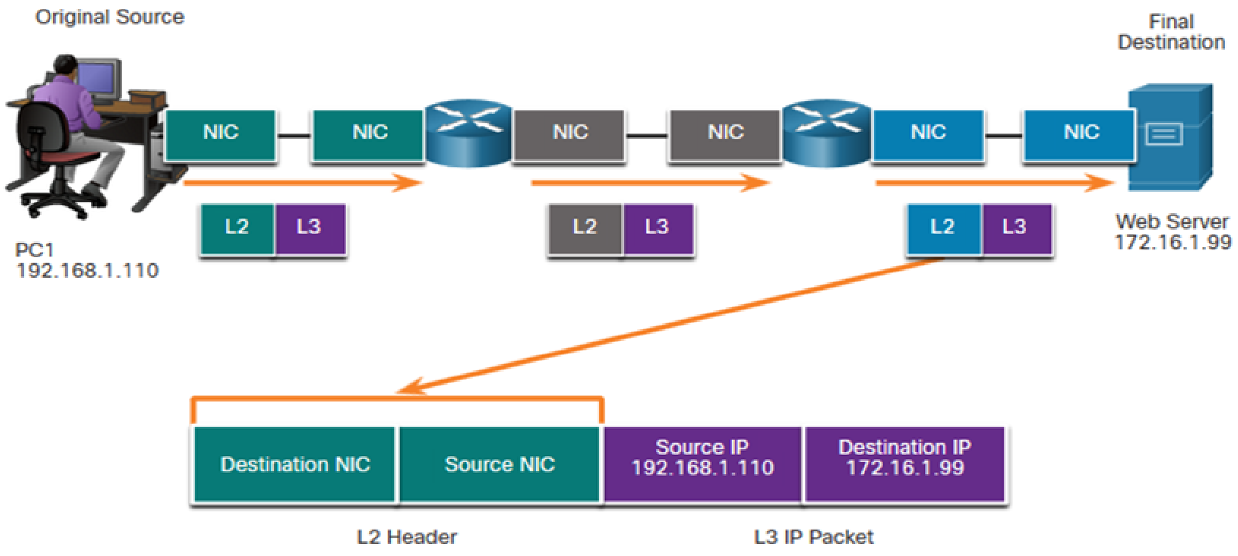
MAC addresses are **physically embedded into the Ethernet NIC and are local addressing.**

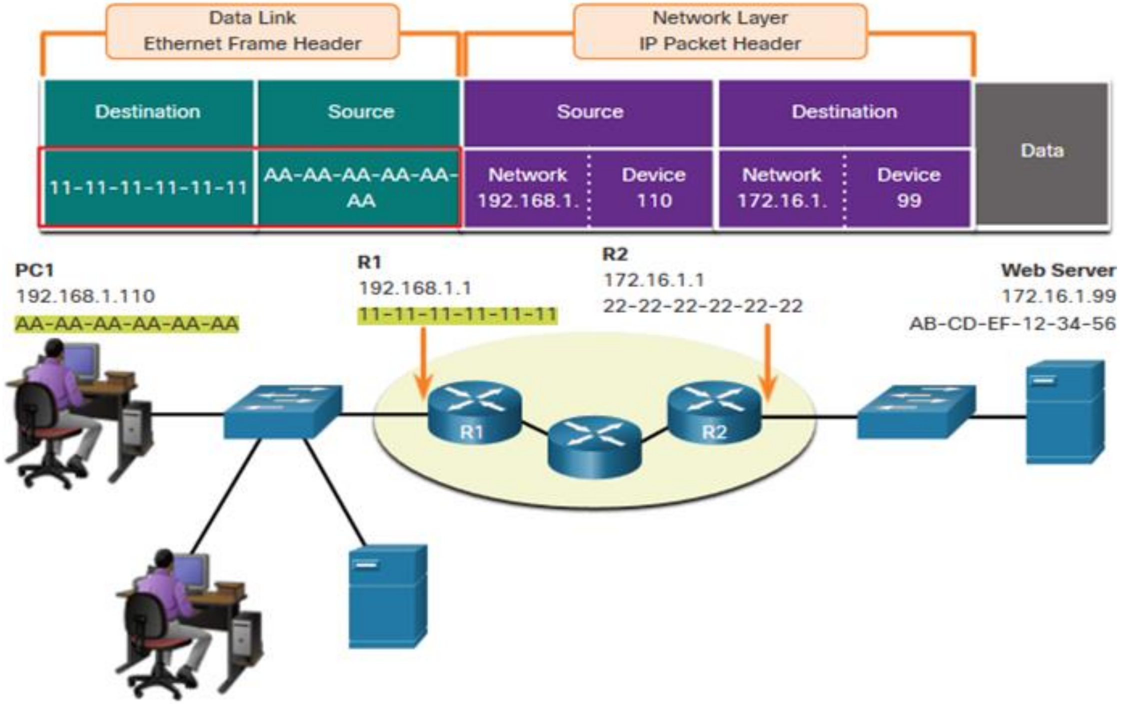
- The Source MAC address will be that of the originator on the link.
- The Destination MAC address will always be on the same link as the source, even if the ultimate destination is remote.

OSI and TCP/IP Networking Models

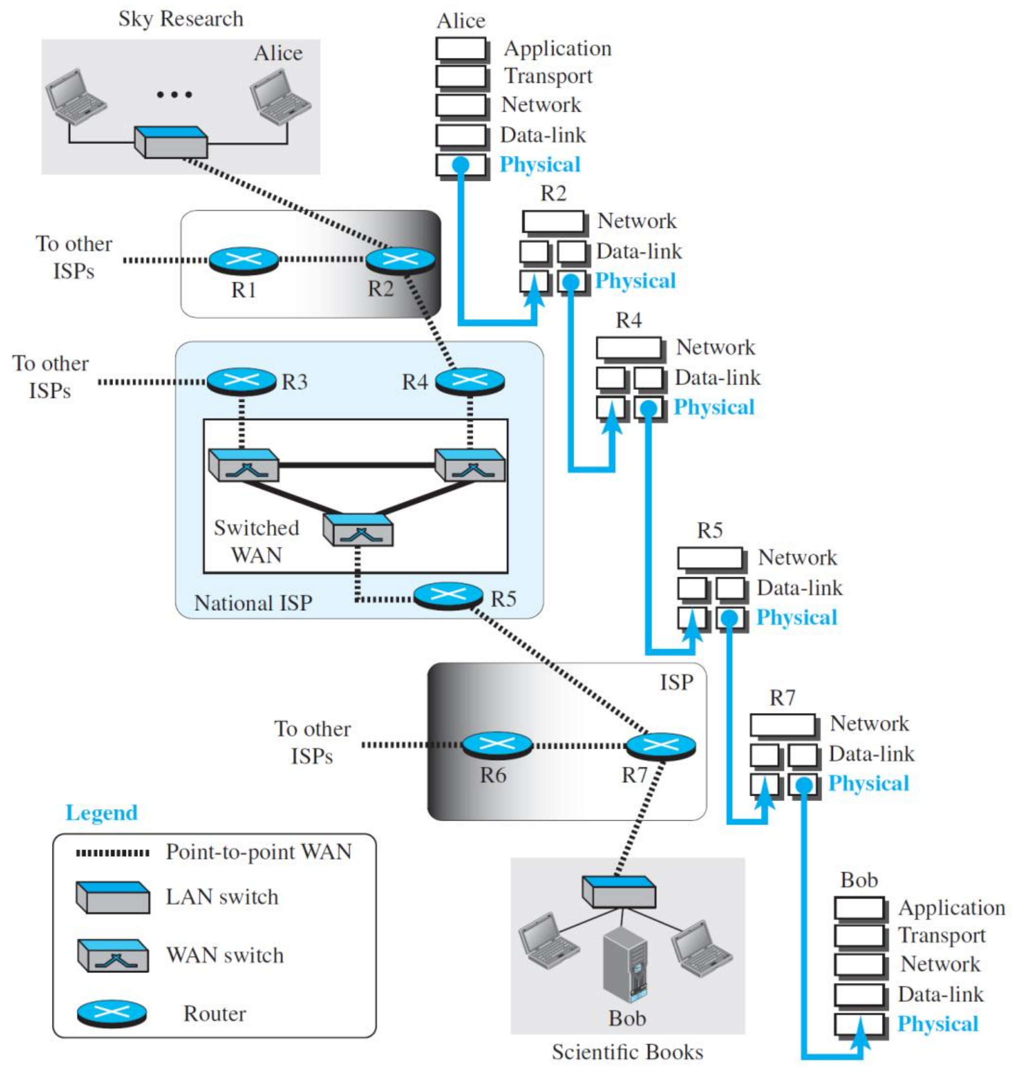
TCP/IP: Datalink Layer Communication (2/2)

Different IP Networks



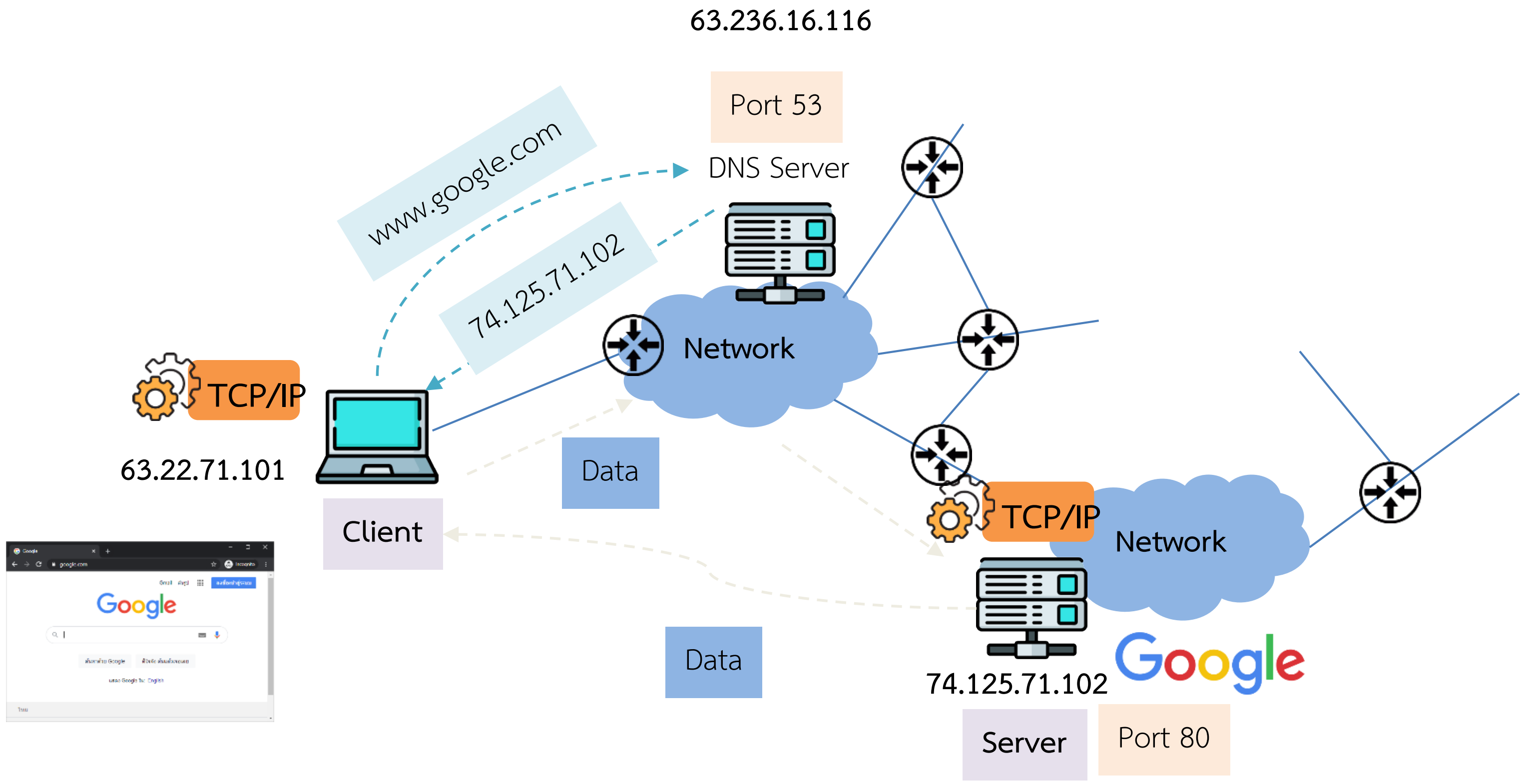


TCP/IP: Physical Layer Communication



Domain Name Service (DNS) (1/2)

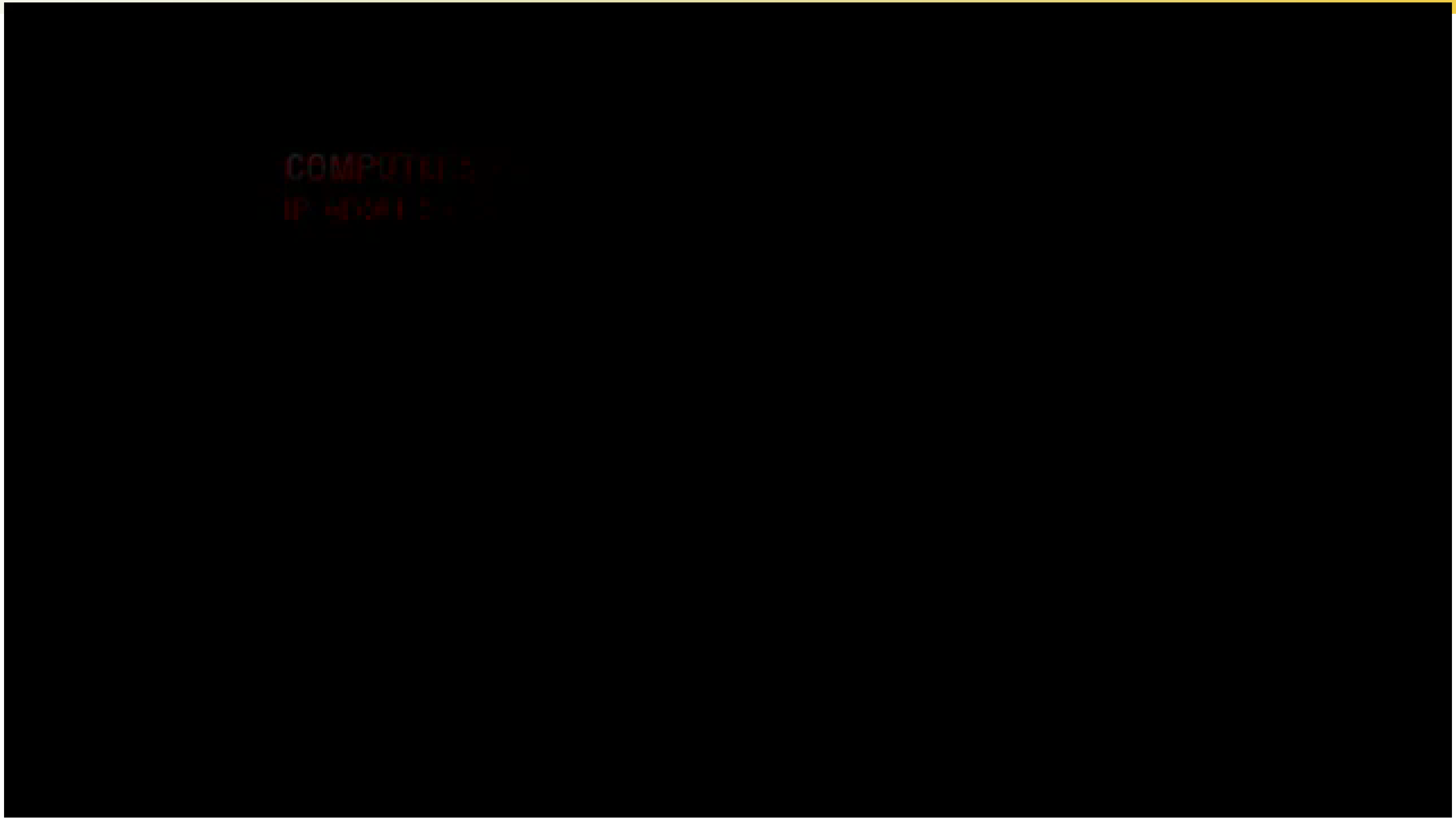
- To convert from domain name to IP address



How DNS Works:

https://www.youtube.com/watch?v=2ZUxoi7YNgs&ab_channel=CIRANEWS

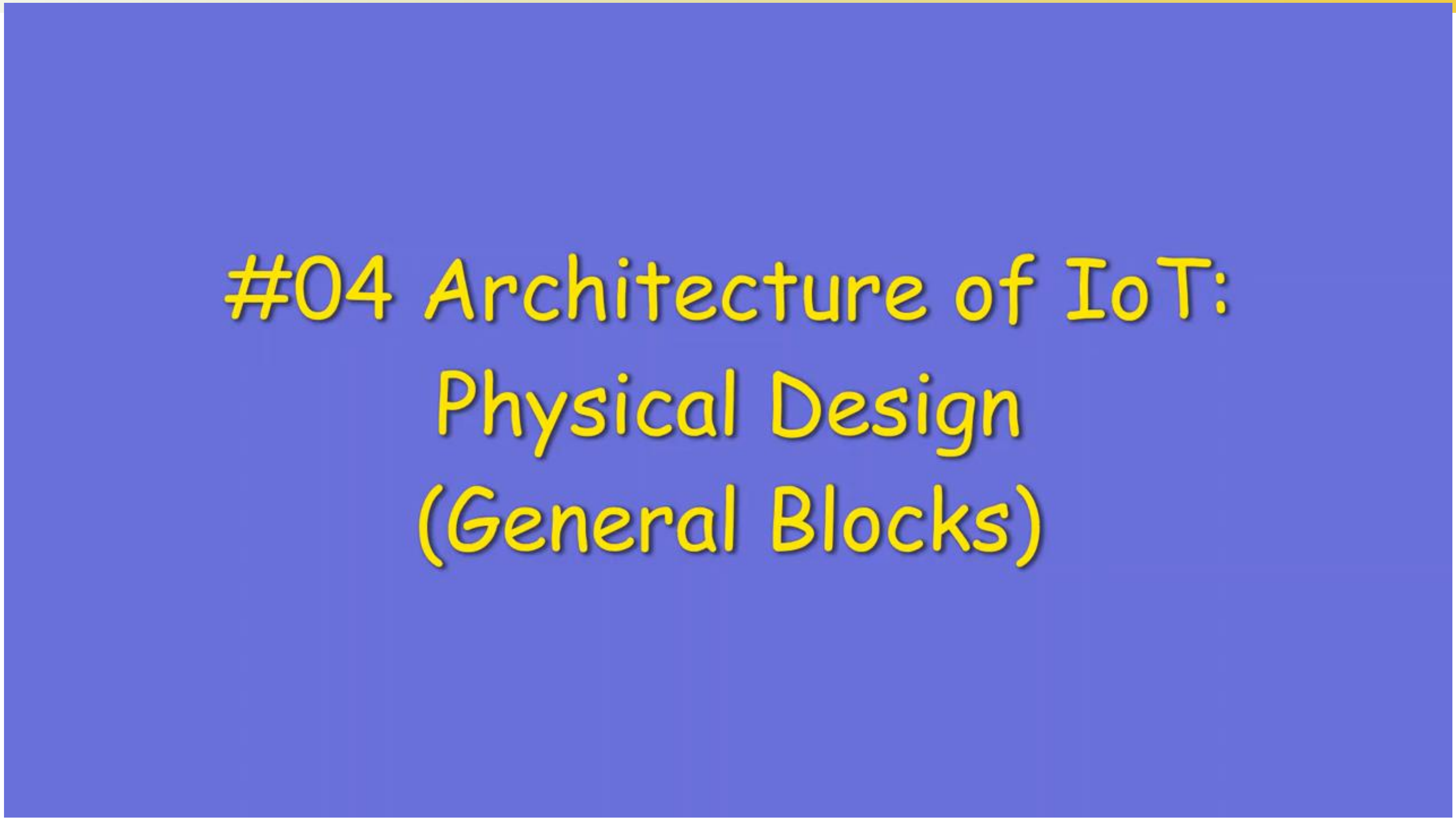
Domain Name Service (DNS) (2/2)



IP Packet Flows (WARRIORS OF THE NET)

https://www.youtube.com/watch?v=RhvKm0RdUY0

Architecture of IoT: Physical Design



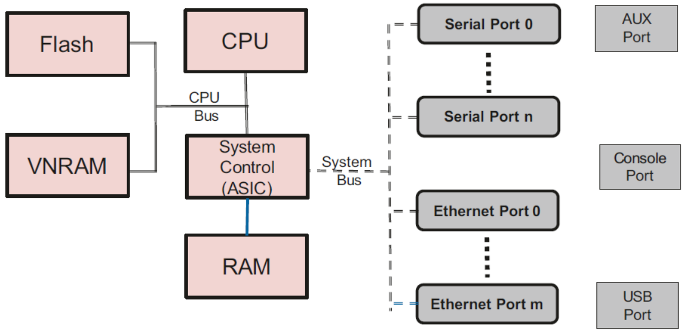
https://www.youtube.com/watch?v=8D7OeoWZEK4

TCP/IP Protocol Stack in IoT

https://www.youtube.com/watch?v=phkdwmvTq8Q

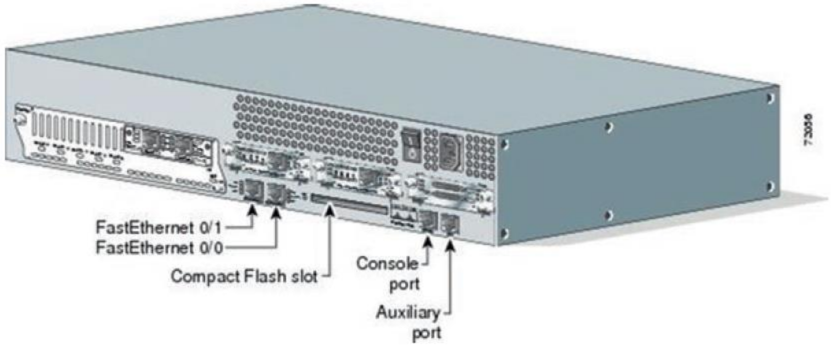
Network Equipment – Router vs IoT Gateway [CISCO]

- An internet of things (IoT) gateway is a physical device or software program that serves as the connection point between the cloud and controllers, sensors and intelligent devices.
- All data moving between IoT devices and the cloud passes through an IoT gateway, which can be either a dedicated hardware appliance or software program. An IoT gateway might also be referred to as an intelligent gateway or a control tier.



Router component	Main function	Volatile/ nonvolatile
CPU	Executes operating system commands: initialization, routing, and switching functions	Nonvolatile
RAM	Stores the instruction and data that CPU needs to execute (considered the working area of memory storage used by the CPU) Stores: "running config" file, routing tables, ARP cache, and buffer	Volatile
ROM	Contains *code for basic functions* to start and maintain the router	Nonvolatile
Flash	Permanently stores the *operating system* (e.g., where a router finds and boots its IOS image)	Nonvolatile
NVRAM	Stores the "startup config" file, holds configuration register software	Nonvolatile
Interfaces/ ports	Routers are accessed and connected to the external world via the interfaces	N/A

IoT Gateway



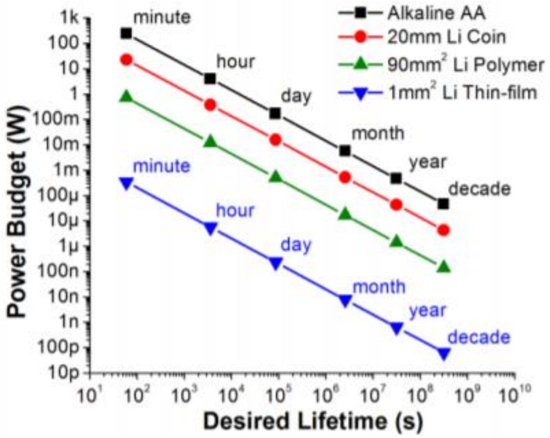
https://www.techtarget.com/iotagenda/definition/IoT-gateway
https://www.cisco.com/c/en/us/products/collateral/wireless/aironet-1815-series-access-points/datasheet-c78-738243.html

A Taxonomy of Constrained Devices [IETF-RFC 7228]

- Maximum code complexity (ROM/Flash).
- Size of run-time state and buffers (RAM).
- Amount of computation feasible in a specific period of time ("processing power").
- Available power resources.
- Management of user interface and accessibility in deployment (ability to set security keys, update software, etc.)

Specifications	Class 0	Class 1	Class 2
RAM	$\ll$ 10kB	$\approx$ 10kB	$\approx$ 50kB
Flash	$\ll$ 100kB	$\approx$ 100kB	$\approx$ 250kB
RTOS support	Devices do not support RTOS	RTOS could be implemented in these devices	RTOS can be operated
Communication protocols	No protocol stack embedded, use gateways for communication	Communicate via lightweight protocols such as CoAP (Constrained Application Protocol). They can communicate with other devices without the help of gateways	Communication protocols such as HTTP are supported
Security Vulnerabilities	Data compromise will result in a basic threat	Data compromise will result in medium threat	Data compromise will result in medium/high threat

Name	Data size	Code size
Class 0	<<10 KB	<<100 KB
Class 1	~10 KB	~100 KB
Class 2	~50 KB	~250 KB



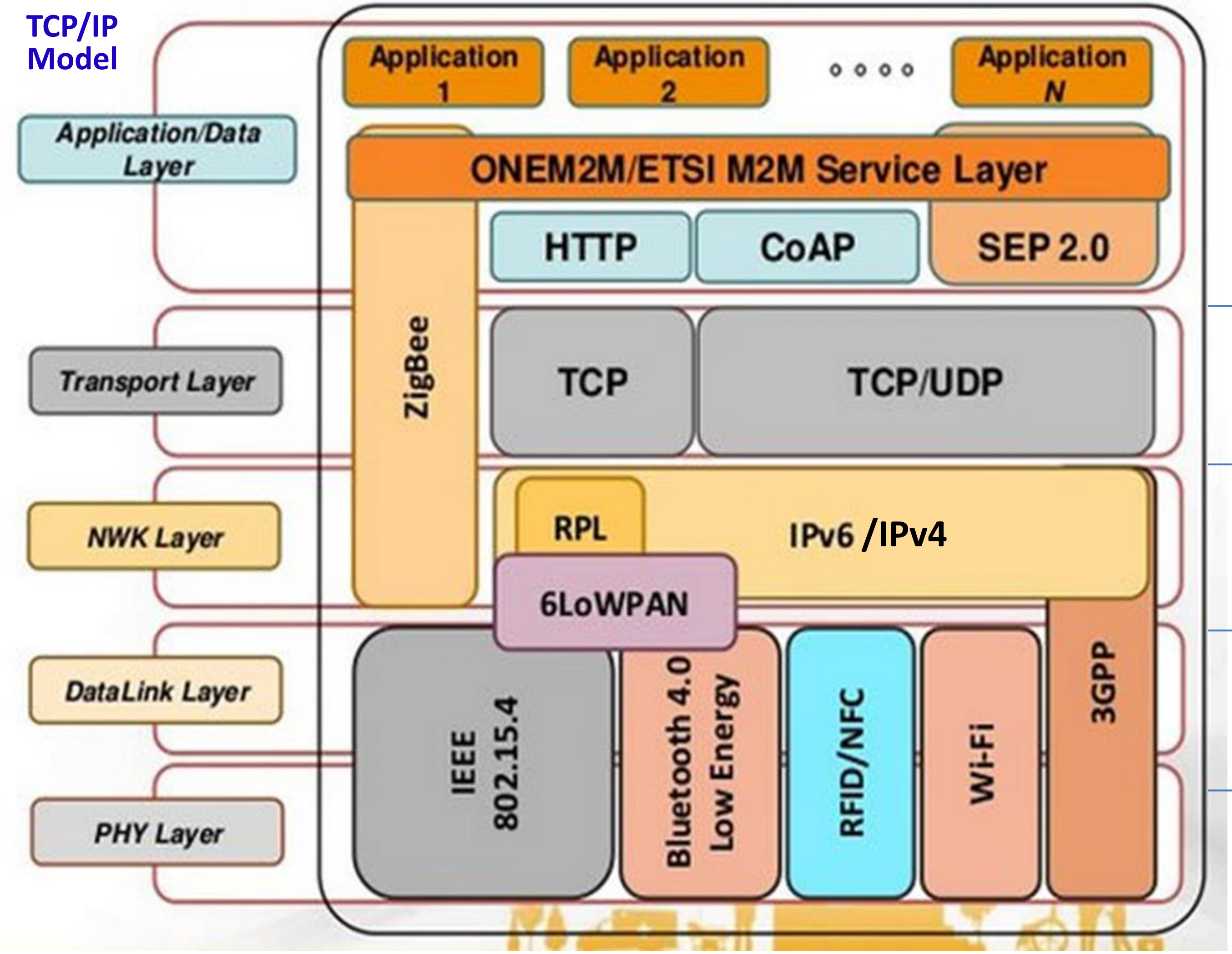
https://tools.ietf.org/html/rfc7228

https://arpi.unipi.it/retrieve/handle/11568/939859/360671/low-end-preprint.pdf

https://www.cisco.com/c/en/us/products/collateral/wireless/aironet-1815-series-access-points/datasheet-c78-738243.html

IoT Protocol Stack

TCP/IP vs OSI (Partial List)



OSI Model [ISO 1984]

Application
Presentation
Session

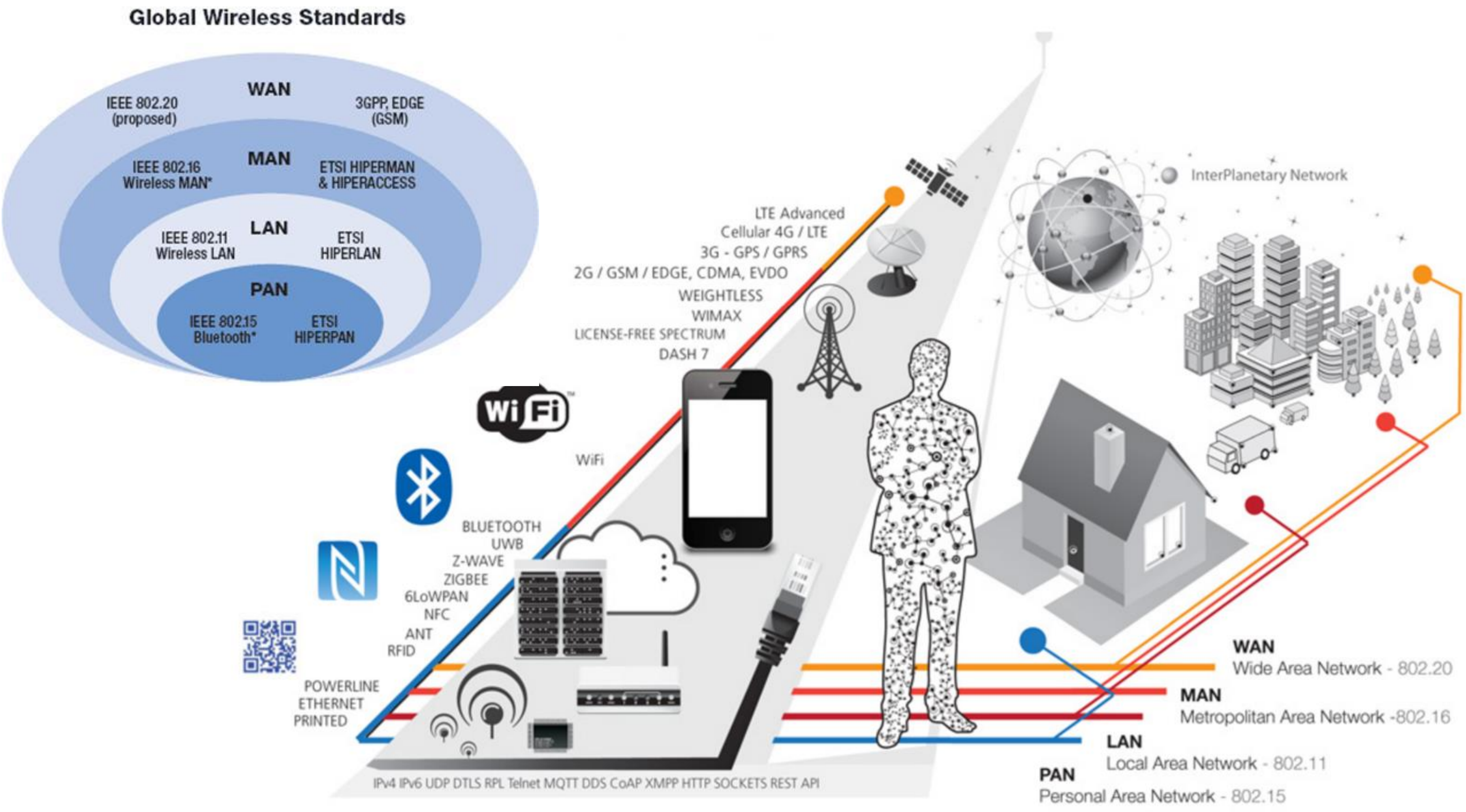
Transport

Network

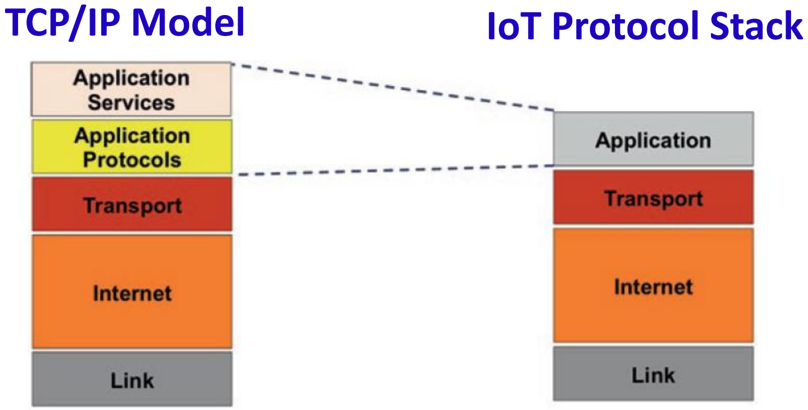
Data Link

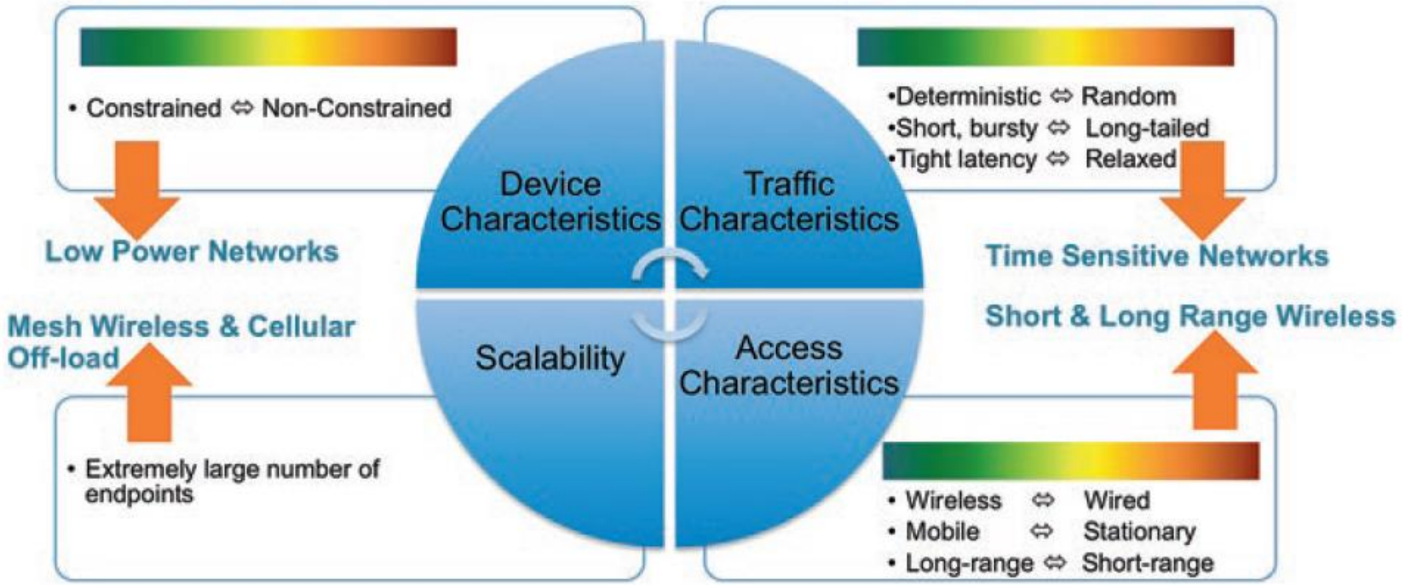
Physical

Wireless Connectivity and Network Size



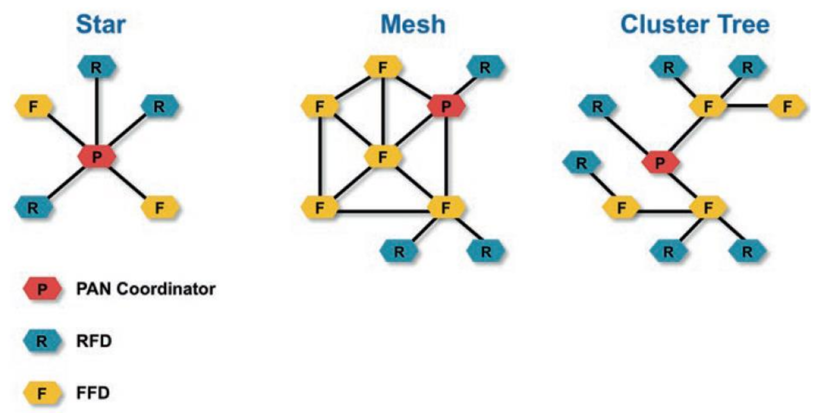
IoT Protocol Stack: Link Layer





IoT Protocol Stack: Link Layer

Personal Area Network (IEEE 802.15.4)



Protocol	Range	Data rate	Topology	Application	Power consumption
IEEE 802.15.4	Up to 1 km	1Mbps to 10Kbps	Mesh	Personal area network/home network	Very low
LoRaWAN	Up to 20 km	Upto 50Kbps	Star	Wide area network	Low
IEEE 802.11ah	Up to 1 km	>100Kbps	Star	Metropolitan block	Medium

Two types of devices in an 802.15.4

FFD: Full-Function Device

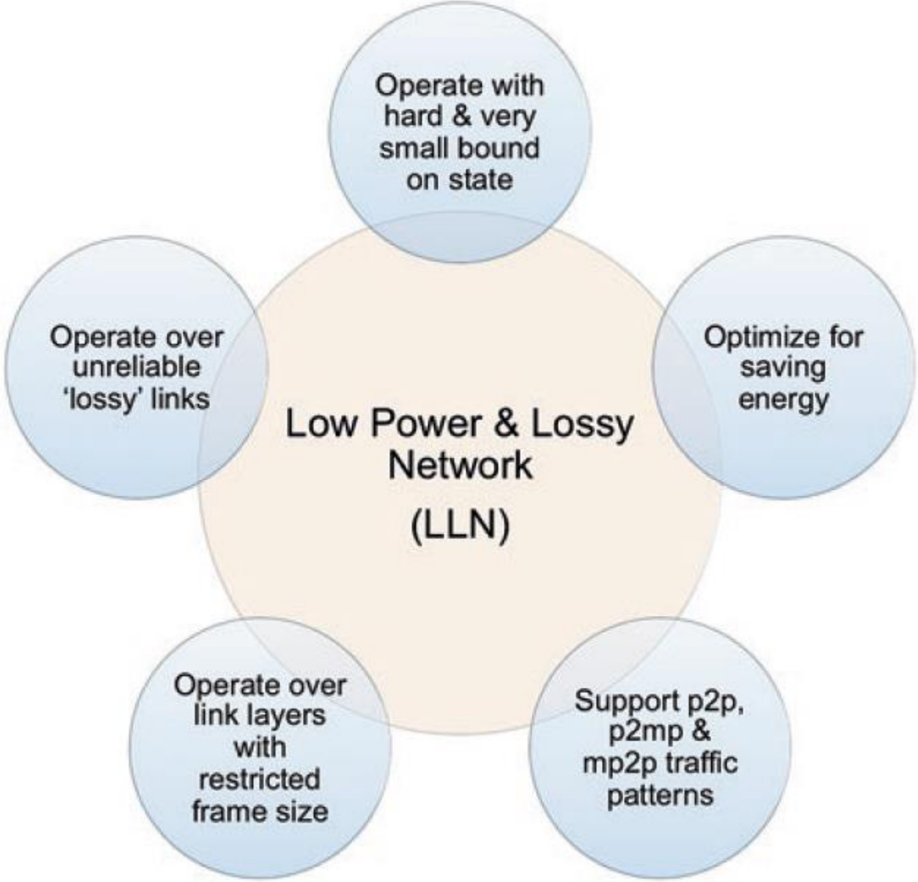
Allows to communicate with other device in the network. It may relay messages.

RFD: Reduced-Function Device

Extremely simple devices are to be embedded into the things.

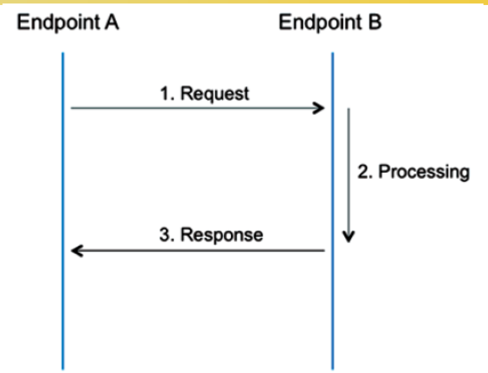
- Many IoT deployments constitute what is referred to as low-power and lossy networks (LLNs).

- The LLNs comprise of a large number (several thousand) of constrained embedded devices with limited power, memory, and processing resources, with a variety of link layer technologies, such as IEEE802.15.4, Bluetooth, Wi-Fi, or Power-Line Communication (PLC) links.



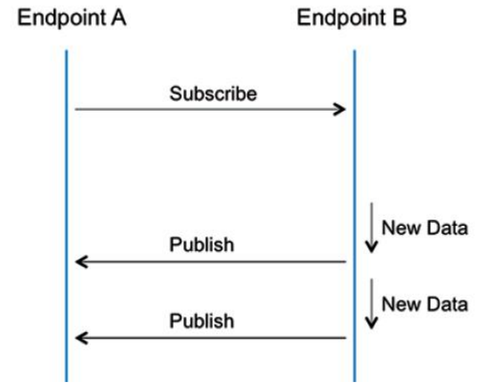
2 Communication Paradigms

- **Request/Response Paradigm**
  - The deployment follows a client-server architecture.
  - The deployment requires interactive communication: both endpoints have information to send to the other side.
  - The receipt of information needs to be fully acknowledged (e.g., for reliability).



- **Publish/Subscribe Paradigm**

  Not all IoT devices can be interactive (like sensors or actuators)
  - Loose coupling between the communicating endpoints, especially when compared with the client-server model.
  - Better scalability by leveraging parallelism and the multicast capabilities of the underlying transport network.



See next Lecture: IoT Application Protocols

IoT Protocol Stack: Application Layer

IoT Application Protocols

Protocol	Functions	Primary use	Transport	Format	SDO
CoAP	REST resource manipulation via CRUD Resource tagging with attributes Resource discovery through RD	LLNs	UDP	Binary	IETF
XMPP	Manage presence Session establishment Data transfer (text or binary)	Instant messaging	TCP HTTP	XML	IETF XSF
MQTT	Lightweight pub/sub messaging Message queuing for future subscribers	Enterprise telemetry	TCP	Binary	OASIS
AMQP	Message orientation, queuing and pub/sub Data transfer with delivery guarantees (at least once, at most once, exactly once)	Financial services	TCP	Binary	OASIS
SIP	Manage presence Session establishment Data transfer (voice, video, text)	IP telephony	TCP, UDP, SCTP	XML	IETF
IEEE 1888	Read/write data into URI Handling time-series data	Energy and facility management	SOAP/ HTTP	XML	IEEE
DDS (RTPS)	Pub/sub messaging with well-defined data types Data discovery Elaborate QoS	Real-time distributed systems (military, industrial, etc.)	UDP	Binary	OMG

Learn more next Lecture: IoT Application Protocols

Technology Gaps on Application Service Layer

Search and discovery capabilities:

- Mechanisms by which devices as well as applications can **automatically discover** each other as well as discover middleware/common services nodes.
- Mechanisms by which applications can **search for devices with specific attributes** (e.g., sensors of particular type) or context (e.g., within a specific distance from a location).
- Mechanisms by which applications can search for data based on attributes (e.g., semantic annotations) or context (e.g., spatial or temporal).

Data encoding, interpretation, and modeling

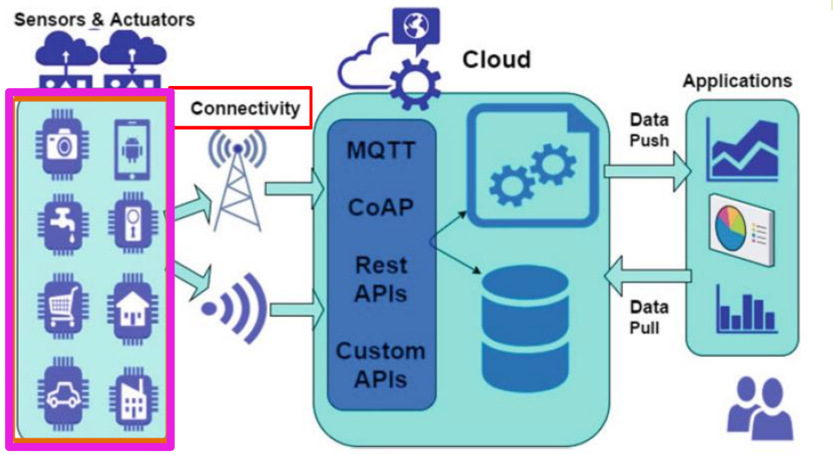
- Mechanisms that render IoT data understandable to applications without a priori knowledge of the data or the devices that produced it.
- Mechanisms that enable application interaction at a high level of abstraction by means of physical/virtual entity modeling.
- Mechanisms that enable data management services to host the semantic description of IoT data that is being handled.
- Framework for defining formal domain-specific semantic models or ontologies, including but not limited to defining an upper-level ontology for IoT.

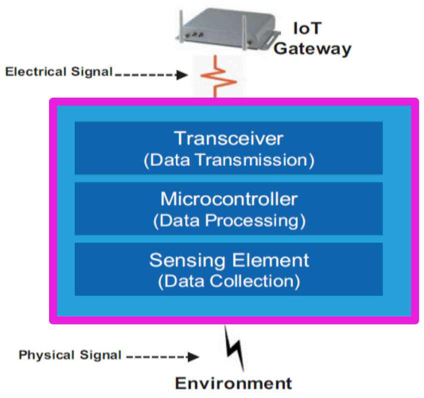
IoT Networking Considerations and Challenges

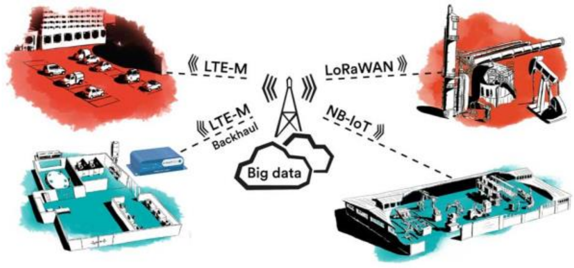
- Scalability: Millions of devices
  - IPv6/IPv4 (DHCP/NAT) addresses
  - tons of data
  - Legacy support: non-IP, specialized devices, multiple vertical solutions
- Range: PAN/LAN/MAN/WAN
- Bandwidth
  - The volume of data each device gathers and transmits
  - The number of devices deployed
  - Whether data is being sent as a constant stream or intermittent bursts, and if any peak periods are notable.
  - Intermittent connectivity: either planned or unexpected link disruptions
- Constrained Devices (lost cost): lossy networks, low bandwidth, small batteries, …
- Need for Real Time: real time streaming data analysis, proactive action, ..
- Different IoT reference architectures: ETSI, ITU, IEEE, NIST, ISO, oneM2M Alliance, ISA, …
- Security: Private/Public Data travels through public networks
  - Authentication, Encryption, Port Protection

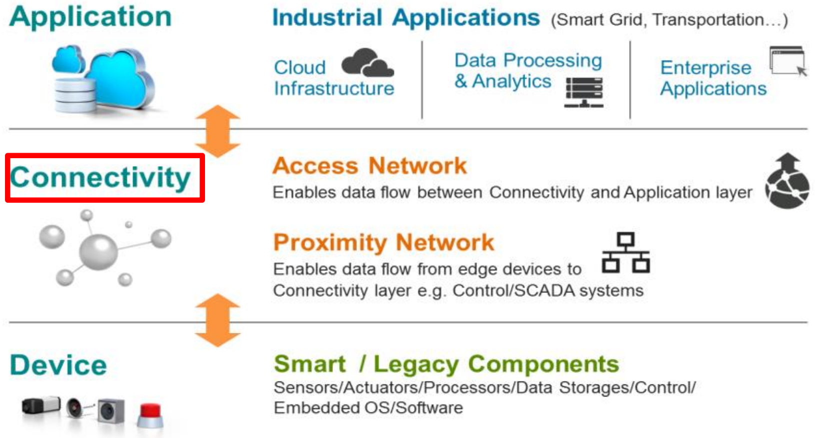
Read more: https://developer.ibm.com/technologies/iot/articles/iot-lp101-connectivity-network-protocols/

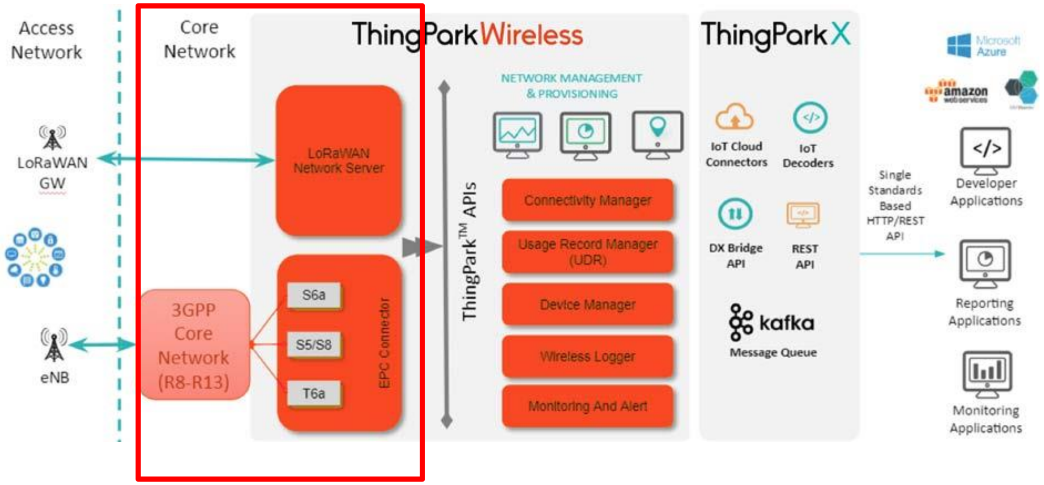
IoT Connectivity vs Access Networks





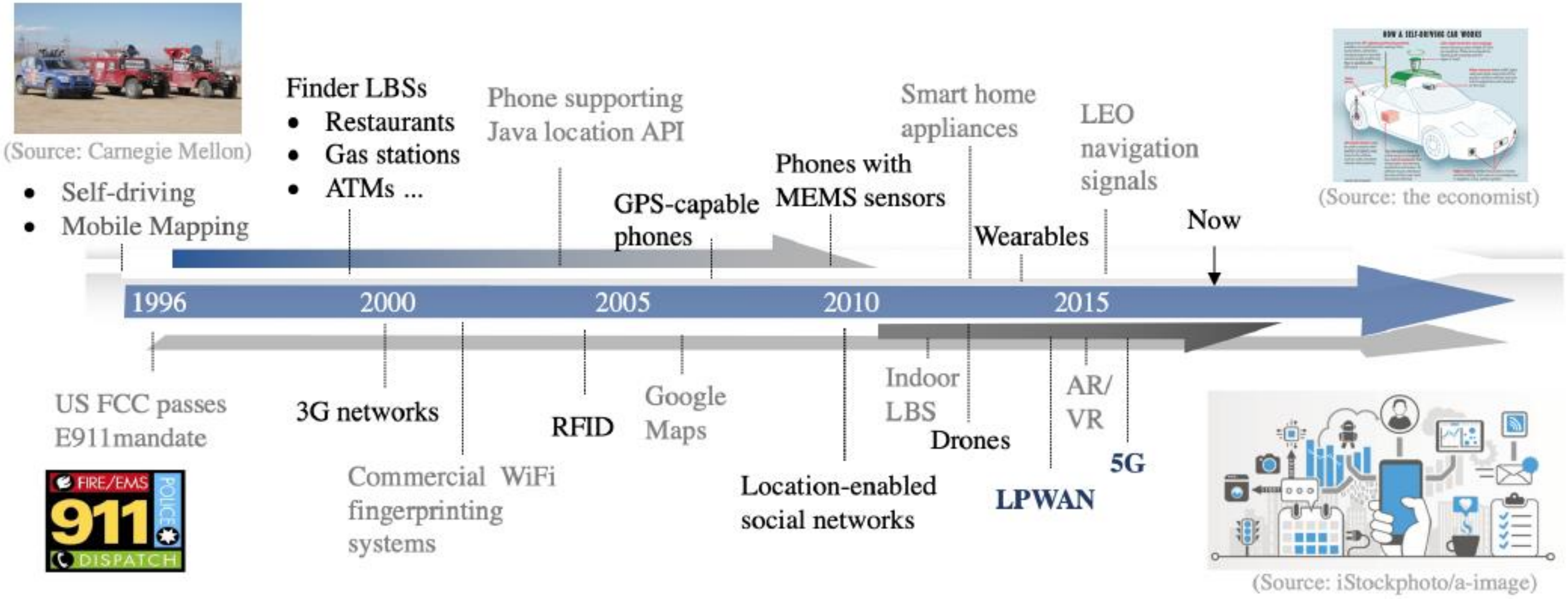






IoT Connectivity vs Access Networks

Location-based Applications

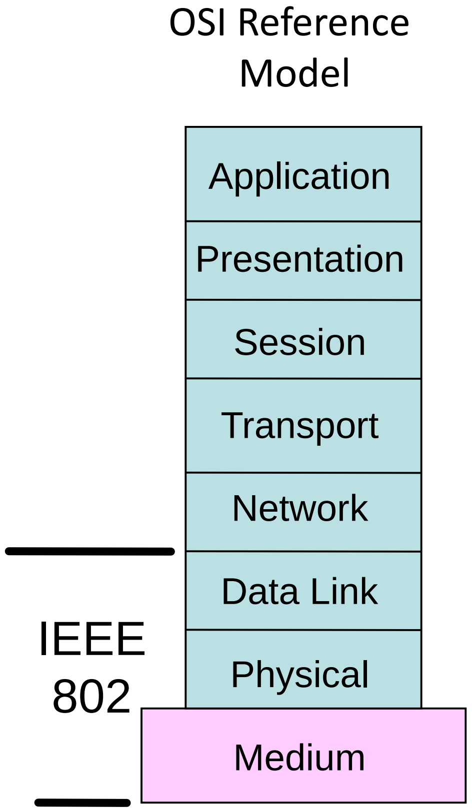


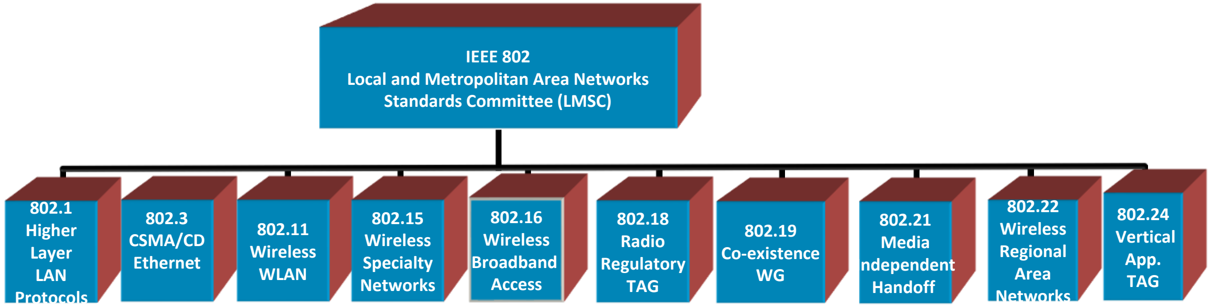
Next-Generation Wireless Technologies

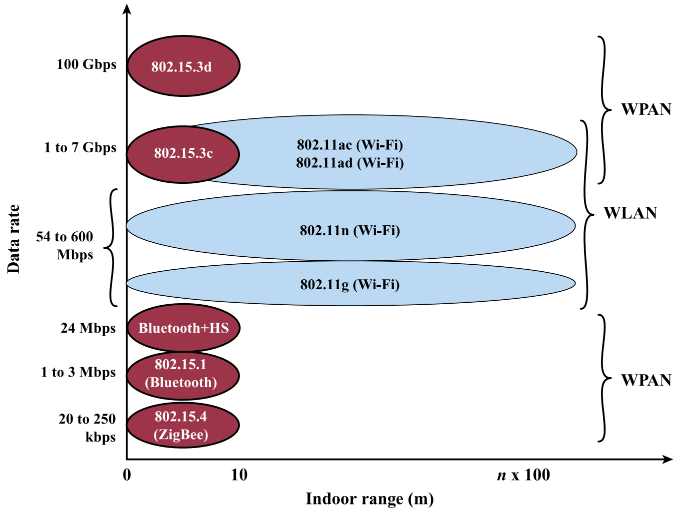
	Frequency	Range	Data rate	Energy consumption	End nodes per gateway	Availability
RFID	125 kHz – 960 MHz	100 m	640 kBit/s	Very low	1	Dedicated infrastructure
	13.56 MHz	10 cm	430 kBit/s	Low	1	Every smartphone
Bluetooth	2.4 GHz	350 m	1 MBit/s	Very low	Unlimited	Every smartphone & some gateways
WiFi 6 CERTIFIED	1, 2.4, 5 & 6 GHz	50 m	10 GBit/s	Medium	250	Every smart-phone & new infrastructure
4G	410 - 5900 MHz	10 km	1 GBit/s	Low	600	Public deployment
5G	2.5 GHz - 49 GHz	500 m	50 GBit/s	Very high	1 million	Limited public deployment
LTE-M	500 - 5900 MHz	100 km	375 kBit/s	Low	Not available	All cellular networks country-specific
NB-IoT	698 - 2170 MHz	40 km	250 kBit/s	Low	ca. 20,000	All cellular networks country-specific
LoRaWAN	490 MHz 868 MHz 915 MHz	50 km	300 Bit/s	Very low	1 million	157 countries
sigfox	433 MHz 915 MHz	50 km	100 Bit/s	Very low	1 million	70 countries
	400 MHz	Global	144Bit/s	Medium	Not applicable	Global

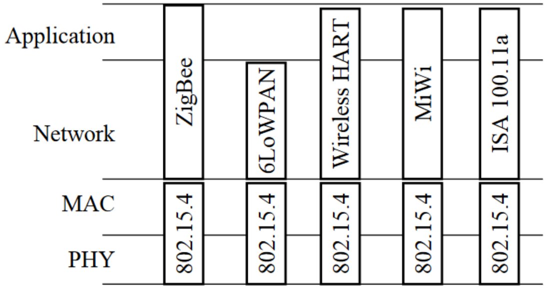
Some specifications may vary by country/region or future releases.

IEEE 802

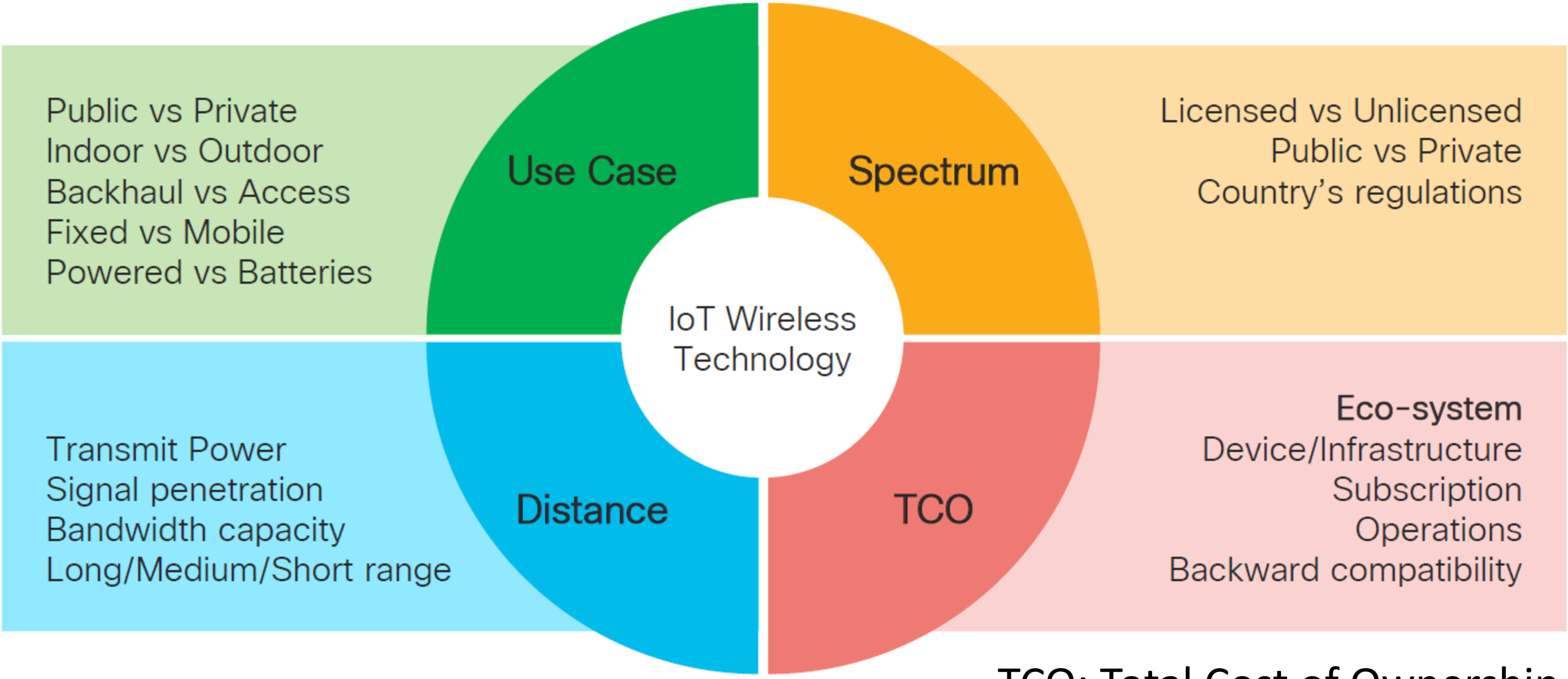

OSI Reference Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
Medium
IEEE 802


IEEE 802
Local and Metropolitan Area Networks
Standards Committee (LMSC)
802.1 Higher Layer LAN Protocols
802.3 CSMA/CD Ethernet
802.11 Wireless WLAN
802.15 Wireless Specialty Networks
802.16 Wireless Broadband Access
802.18 Radio Regulatory TAG
802.19 Co-existence WG
802.21 Media Independent Handoff
802.22 Wireless Regional Area Networks
802.24 Vertical App. TAG


Data rate
100 Gbps
1 to 7 Gbps
54 to 600 Mbps
24 Mbps
1 to 3 Mbps
20 to 250 kbps
802.15.3d
802.15.3c
802.11ac (Wi-Fi)
802.11ad (Wi-Fi)
802.11n (Wi-Fi)
802.11g (Wi-Fi)
Bluetooth+HS
802.15.1 (Bluetooth)
802.15.4 (ZigBee)
WPAN
WLAN
WPAN
0
10
n x 100
Indoor range (m)


Application
Network
MAC
PHY
ZigBee
6LoWPAN
Wireless HART
MiWi
ISA 100.11a
802.15.4
802.15.4
802.15.4
802.15.4
802.15.4

Industrial IoT Wireless Selection Criteria



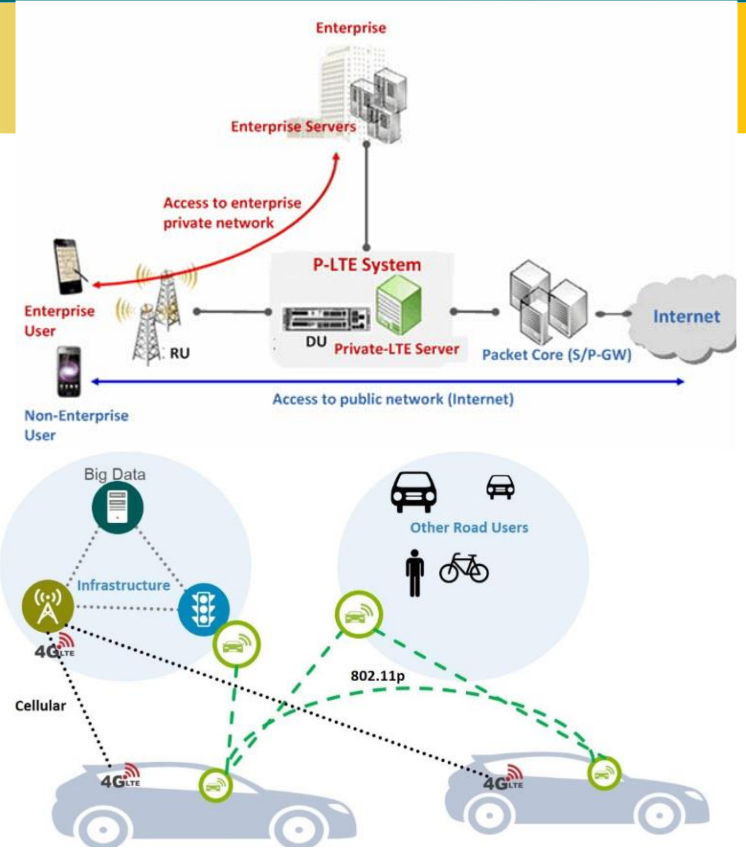
TCO: Total Cost of Ownership

IoT Connectivity vs Access Networks

Use Cases for IoT Wireless

From bits/sec to gigabits/sec

Industries	Use Cases	Wireless Technologies Access (A) or Backhaul (B)
Manufacturing, Warehouse, Distribution Center	Industrial automation, industrial security, plant efficiency, workforce enablement	LoRaWAN (A), Wi-Fi(A/B), 4G (B), 5G (B)
Transportation	Passenger experience, data operations, operational efficiency, safety and compliance, traffic operations, roadway safety, sustainable mobility, sensor modernization	LoRaWAN (A), Wi-Fi (A/B), DSRC (A) 4G (B), 5G (A/B)
Cities	Cities operations, public safety and security, citizen services, economic sustainability	LoRaWAN (A), Resilient Mesh (A), Wi-Fi (A/B), 4G (B), 5G (B)
Mining	Field operations, industrial security, workforce enablement	LoRaWAN, (A) WirelessHart (A), ISA100.11a (A), Wi-Fi (A/B), 4G (B), p-LTE (A/B), 5G (B)
Oil & Gas	Plant and field operations, industrial security, workforce enablement	LoRaWAN, (A) WirelessHart (A), ISA100.11a (A), Wi-Fi (A/B), 4G (B), p-LTE (A/B), 5G (B)
Utilities	Connected substations, distribution grid management, workforce enablement, grid safety, production plants	LoRaWAN (A), Resilient Mesh (A), Wi-Fi (A/B), 4G (B), P-LTE (B), 5G (B)



P-LTE: Private 4G
LTE: Long Term Evolution

https://www.cisco.com/c/m/en_us/solutions/industries/portfolio-explorer.html
https://en.wikipedia.org/wiki/List_of_mobile_phone_generations
https://en.wikipedia.org/wiki/IEEE_802.11
https://www.zdnet.com/article/what-is-v2x-communication-creating-connectivity-for-the-autonomous-car-era/
https://www.fcc.gov/media/radio/public-and-broadcasting

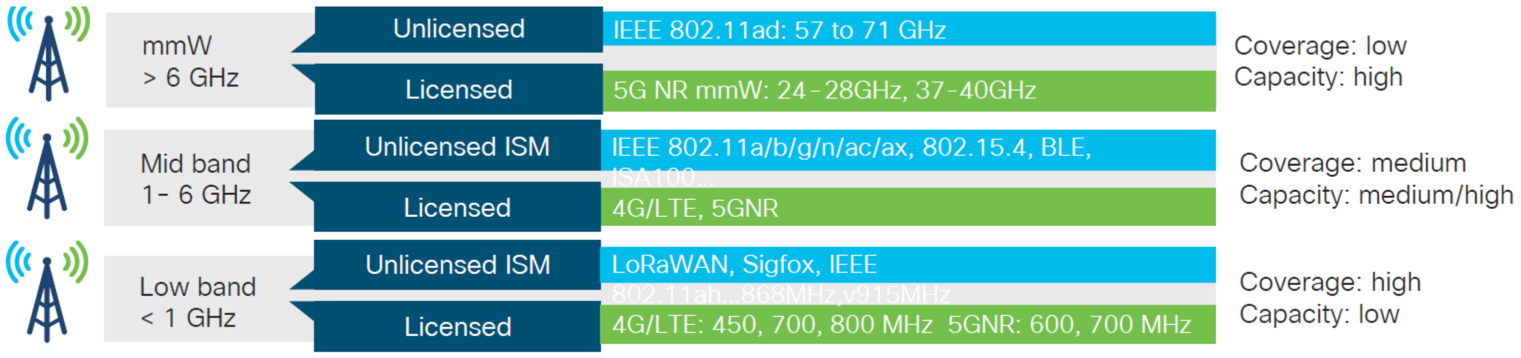
IoT Connectivity vs Access Networks

Spectrum is a Scarce Resource

Managed by World organizations/Countries strongly regulated: Transmit power, duty cycle…

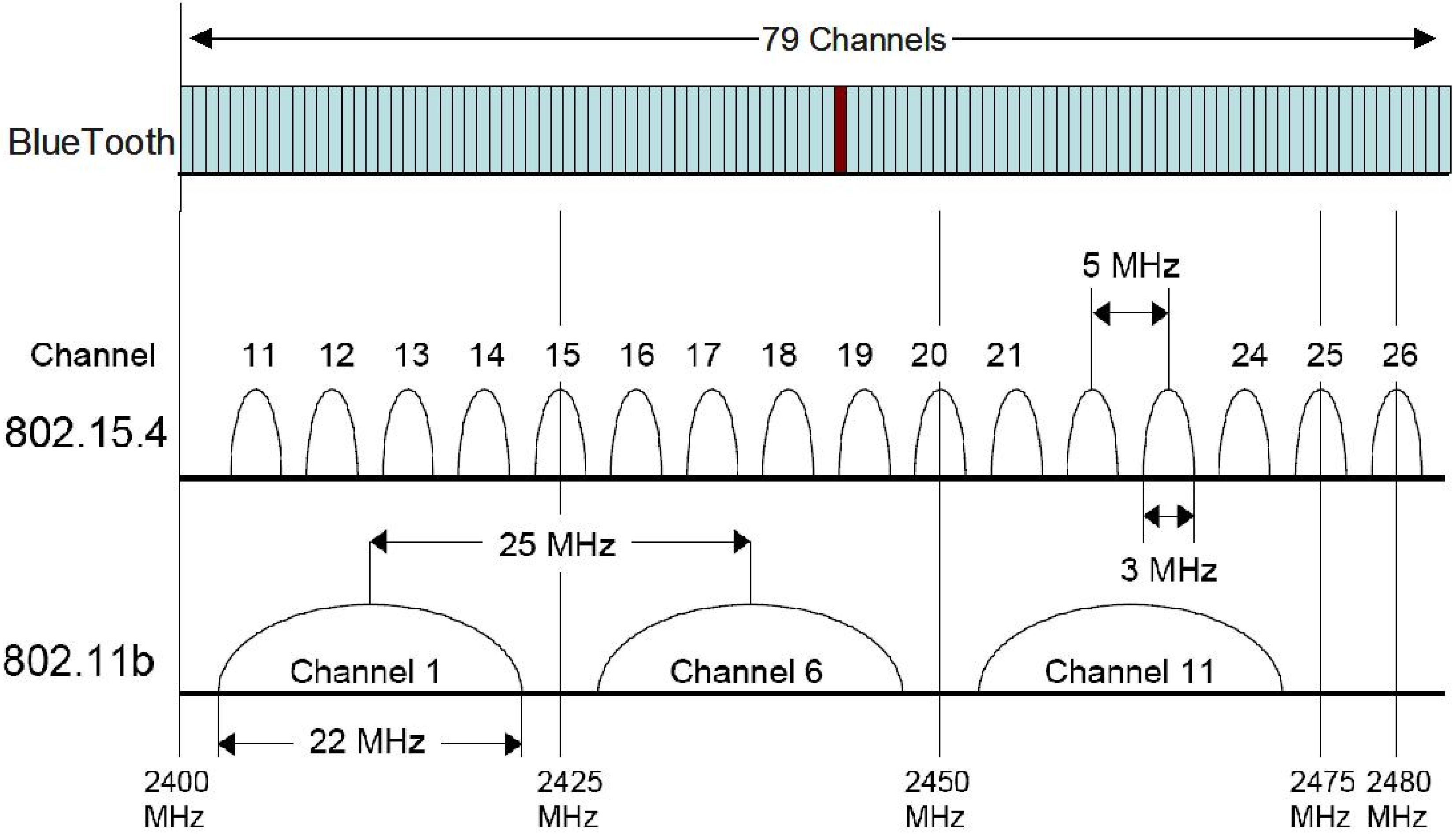
Spectrum types in IoT Wireless

- Unlicensed: also refer as ISM (Industrial, Scientific and Medical) bands, generally free of charge, public, and private infrastructures, but regulated.
  - Shared between technologies; co existence definition in specifications
- Licensed: dedicated to SP (public services) or industries (private), not free, may be reallocated.
  - Introducing Shared licensed model.



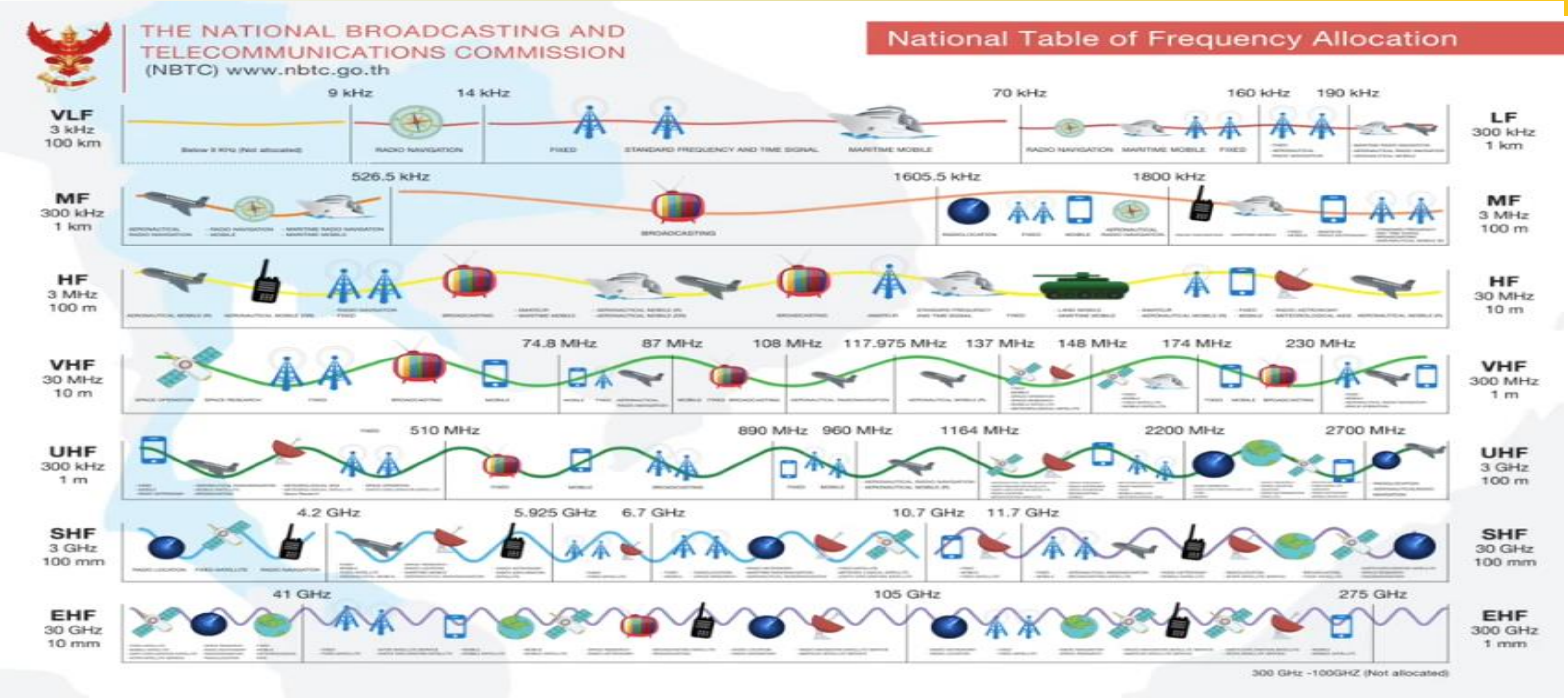
https://en.wikipedia.org/wiki/ISM_band

Example: ISM Band



IoT Connectivity vs Access Networks

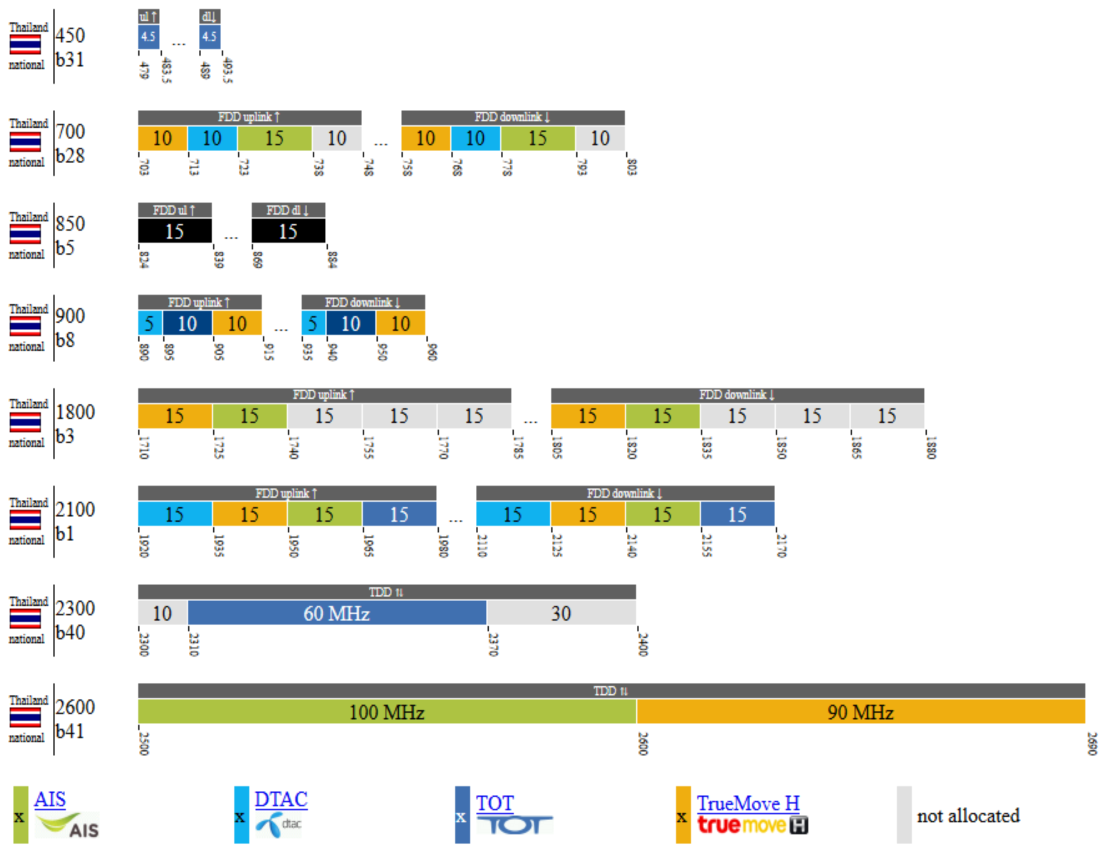
Shared Licensed: Radio-Frequency Spectrum

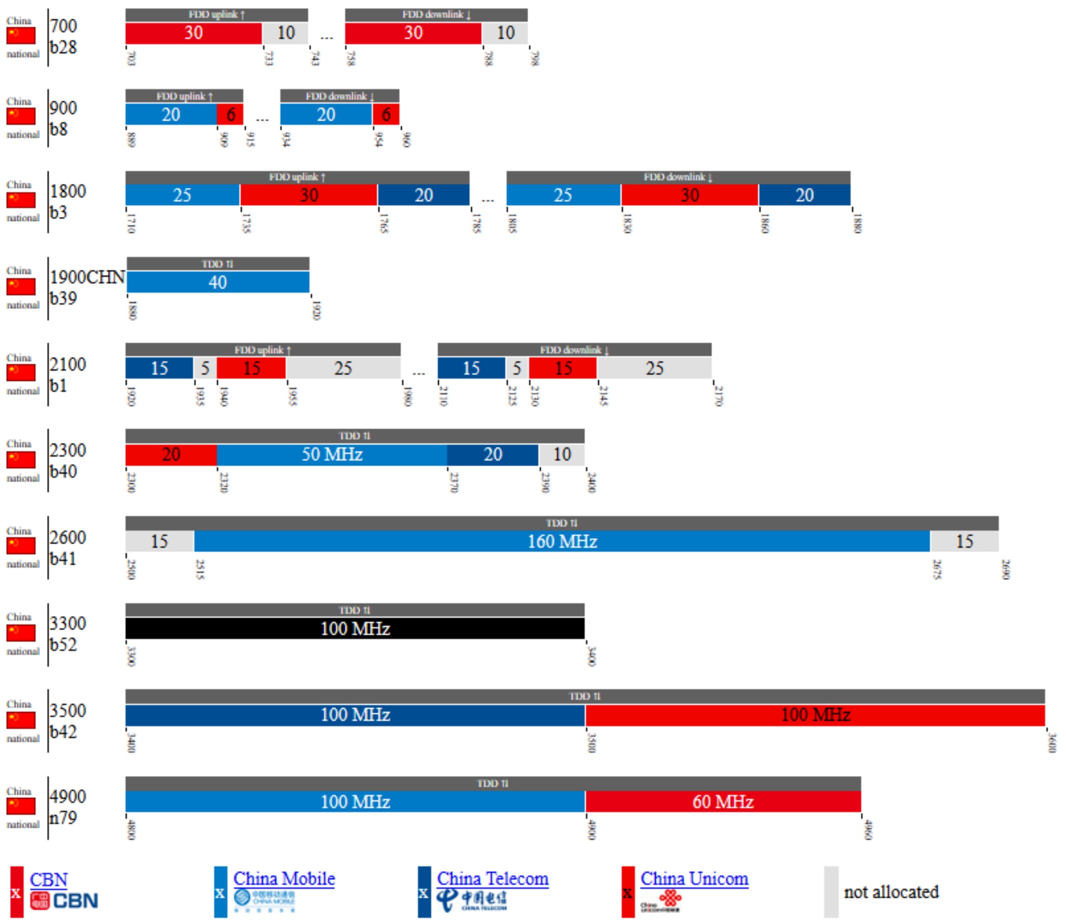


https://www.youtube.com/watch?v=VzD-L24vg4U

IoT Connectivity vs Access Networks

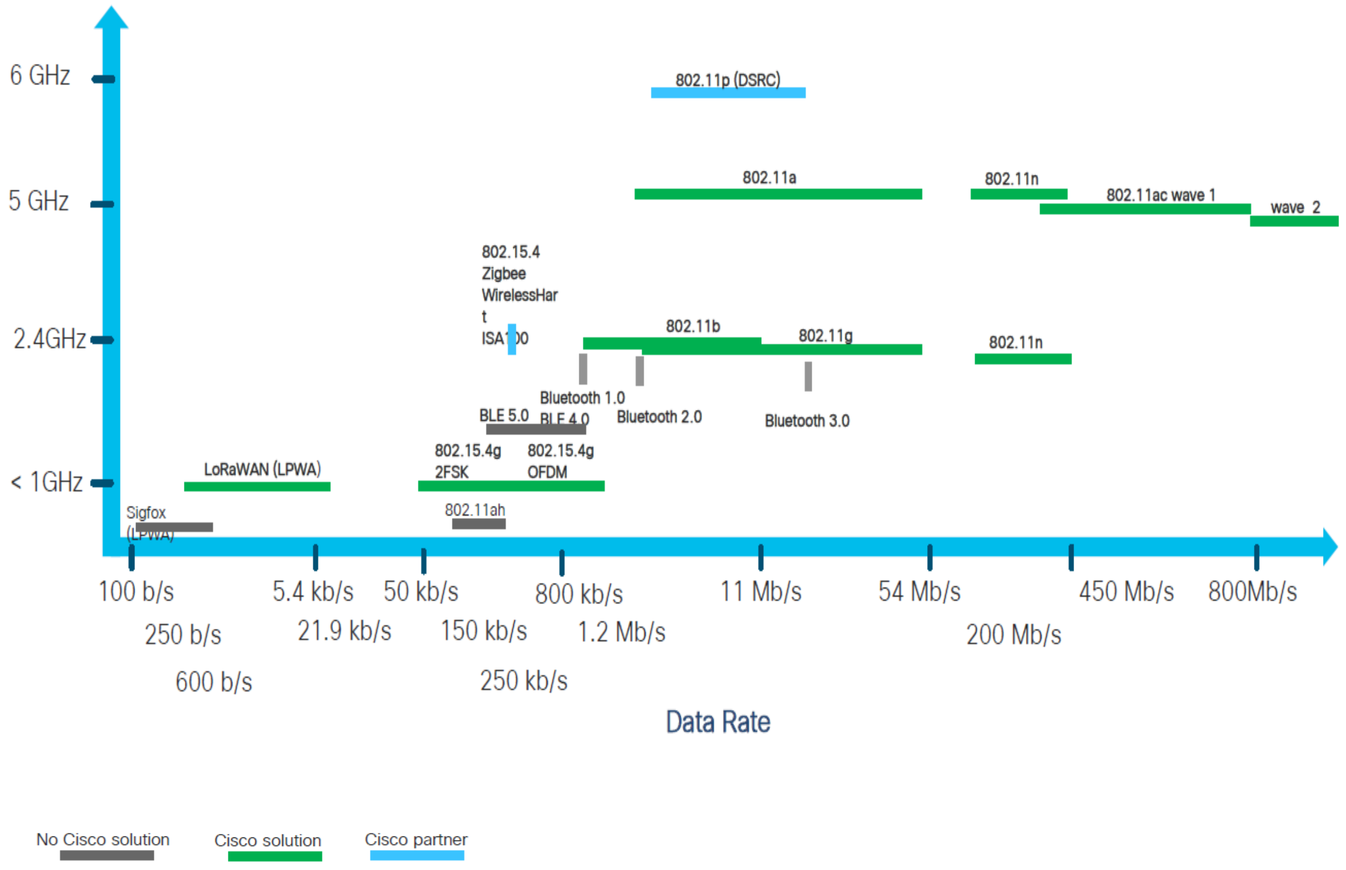
Shared Licensed: Global Mobile Frequencies





https://www.spectrummonitoring.com/frequencies.php

Unlicensed Bands: IoT Wireless Technologies e.g. CISCO



- In addition to spectral auctions, spectrum can be set aside in specific frequency bands that are **free to use** with a license according **to a specific set of etiquette rules**.
- The purpose of these unlicensed bands is to encourage innovation and low-cost implementation.
- Many extremely successful wireless systems operate in unlicensed bands, including **wireless LANs, Bluetooth, and cordless phones**.
- A major difficulty is that they can be killed by their own success.
  - If many unlicensed devices in the same band are used in close proximity, they generate much interference to each other, which can make the band unusable.

IoT Connectivity vs Access Networks

Licensed/Unlicensed Bands in IoT 1/2

Technology	Standard	Distance	Data rate	Freq. Band	Licensed	Indoor/ outdoor	Mobility	Topology	Latency
3G	IMT-2000	35 km	384 kb/s	1.25 MHz/ 5 MH	Both	Indoor /Outdoor	90 km/h	-	25ms
4G	Single unified LTE + WiMax	100 km	3Gb/s, 1,5Gb/s	1.8-2.6 GHz	Both	Indoor /Outdoor	110 km/h	-	15ms
LTE	3GPP Rel.8	75 km	30 0Mb/s(DL), 75 Mb/s(UL)	2.5 GHz, 5 GHz, 10 GHz	Both	Indoor /Outdoor	350km/h	Star	15ms
LTE-A	3GPP Release 10, 11, 12	200 km	1Gb(DL), 500Mb/s(UL)	2.5 GHz, 5 GHz, 10 GHz, 15 GHz, 20 GHz	Both	Indoor /Outdoor	350km/h-500km/h	Point to point	10ms
LTE-A Pro	3GPP Release 13 and beyond	150km	higher data rate beyond 3 Gbps	licensed (400 MHz to 3.8 GHz) and unlicensed (5GHz) spectrum	Both	Indoor /Outdoor	500km/h	End-to-End Network Slicing for Multiple	2ms
5G	Single unified, 4G + WWWW	300km	10-50 Gb/s	1.8-2.6 GHz– 30-300 GHz, UP 60GHz	Both	Indoor /Outdoor	500 km/h	End-to-End Network Slicing for Multiple	1ms
LoRa	Propietary	22 Km	290 bps-50 kbps	868 MHz and 900 MHz	Unlicensed	Indoor/Outdoor	—	Star	—

Technology	Standard	Distance	Data rate	Freq. Band	License	Indoor/ outdoor	Mobility	Topology	Latency
IEEE 802.16e (WiMax)	3GPP	50 km	15 Mbps 90 Mbps	2.4 GHz and 2.4835 GHz or between 5.75 GHz and 5.850 GHz	Licensed	Indoor/ Outdoor	-	Point-to-multipoint, mesh	30ms
IEEE 802.16m (Mobile WiMax)	3GPP	30-100 km	120 Mbit/s downlink/ 60 Mbit/s	up to 100MHz	Both	Indoor /Outdoor	120km/h	Point-to-multipoint, mesh	27.5

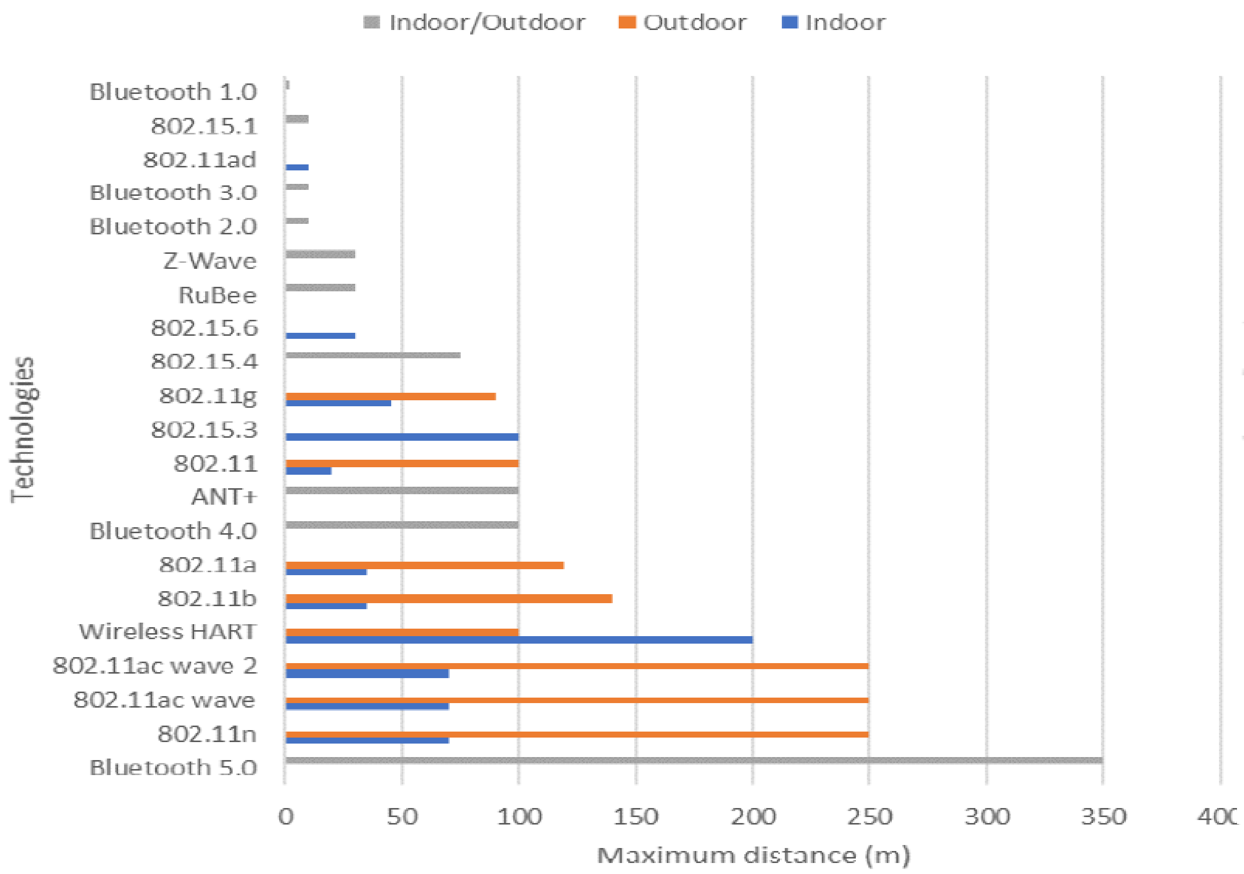
Licensed/Unlicensed Bands in IoT 2/2

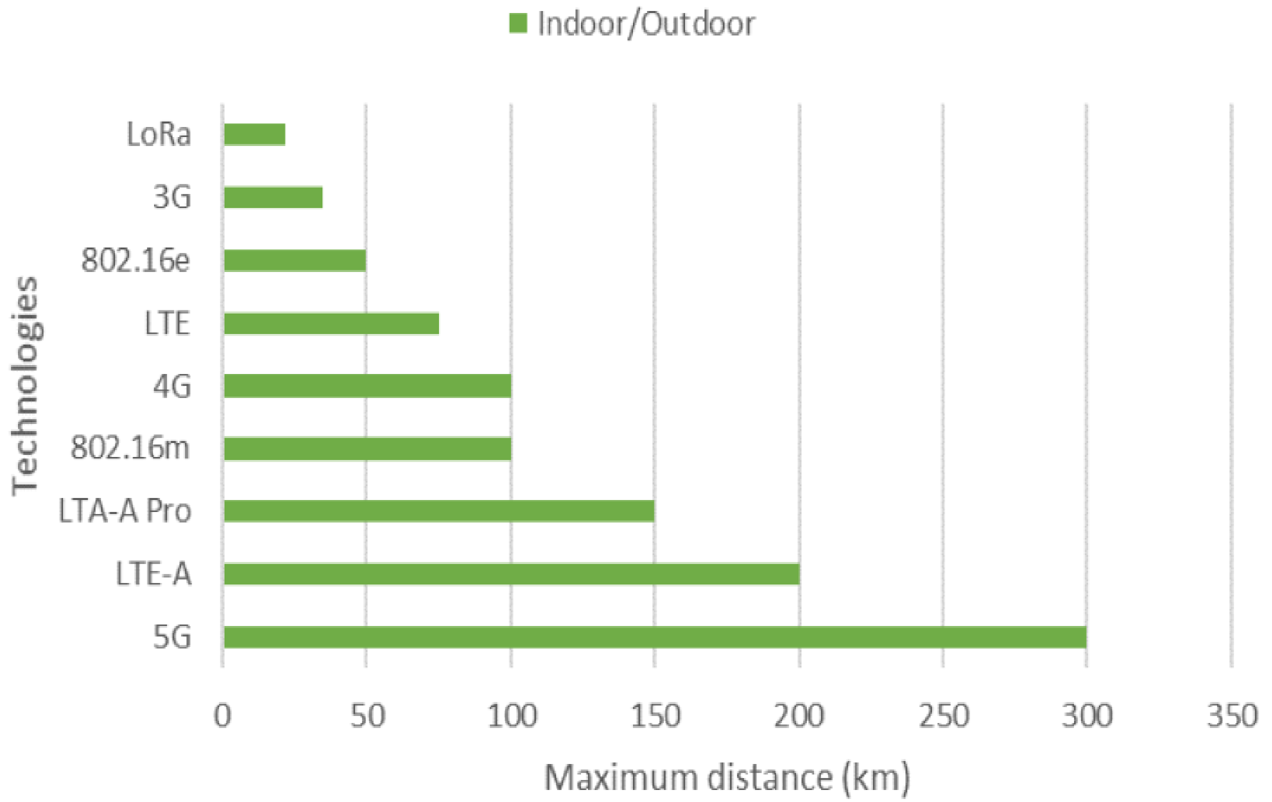
Technologies	Distance	Bitrate	Freq. bands	License	Int/Ext	Energy consumption	Capability
IEEE 802.15.1	10 m	1 Mbps	2.4GHz	unlicensed	Indoor/ Outdoor	0-10 dBm	8 actives 255 park mode
Bluetooth 1.0	2m	721 kbps	2.4GHz	unlicensed	Indoor/ Outdoor	Not specified	—
Bluetooth 1.2	—	721 Kbps	2.4GHz	unlicensed	Indoor/ Outdoor	Not specified	8 actives 255 park mode
Bluetooth 2.0	10 m	2.1 Mbps	2.4GHz	unlicensed	Indoor/ Outdoor	Not specified	8 actives 255 park mode
Bluetooth 3.0	10 m	24 Mbps	2.4GHz	unlicensed	Indoor/ Outdoor	Not specified	8 actives 255 park mode
Bluetooth 4.0	100 m	1 Mbps	2.4 GHz	Unlicensed	Indoor/ Outdoor	Not specified	8 actives 255 park mode
Bluetooth 5.0 [82]	350 m	1-2-3 Mbps	2.4 GHz	Unlicensed	Indoor/ Outdoor	1-100 mW	8 actives 255 park mode
IEEE 802.15.3	10-100m	110-480 Mbps	3.1-10.6 GHz	Licensed	Indoor	-41.3dBM/MHz	—
IEEE 802.15.4	10-75 m	20 kbps/ 40 kbps/ 250 Kbps	868/915 MHz; 2.4 GHz	Licensed/ Unlicensed	Indoor/ Outdoor	-25 dBm / -0 dBm	254
IEEE 802.15.6 [83]	10 m 30 m	480 Mbps 50 Mbps	3.1-10.6 GHz	Unlicensed	Indoor	—	—
ANT+ [83]	100 m/20m	1 Mbps	2.4 GHz	Unlicensed	Indoor/ Outdoor	—	200
RuBee (IEEE 1902.1) [83]	30 m	1024 bps	132 KHz	Licensed	Indoor/ Outdoor	—	—
Insteon [83]	—	2880 bps	900 MHz	Unlicensed	Indoor/ Outdoor	—	—
Z-Wave [83]	30 m	9.6 kbps /40 kBps	868/915 MHz 2.4 GHz	Licensed/ Unlicensed	Indoor/ Outdoor	Low	232
WirelessHART [84]	75 m/ 100 m outdoor/ 200 m line-of-sight	250 Kbps	2.4 GHz	Unlicensed	Indoor/ Outdoor	10 dBm	160

Technologies	Indoor/ Outdoor	Bitrate	Freq. bands	License	Bandwidth	Modulation	MIMO
IEEE 802.11	20m /100m	2 Mbps	2.4GHz	Unlicensed	20 MHz	FHSS and DSSS	—
IEEE 802.11b	35m/ 140m	11 Mbps	2.4GHz	Unlicensed	20 MHz	HR-DSSS	—
IEEE 802.11a	35m/ 119m	54 Mbps	5GHz	Unlicensed	20 MHz	OFDM	—
IEEE 802.11g	45m/ 90m	54 Mbps	2.4 GHz	Unlicensed	22 MHz	OFDM/ DSSS/ CCK	—
IEEE 802.11n	70m/ 250m	600 Mbps	2.4 GHz/ 5 GHz	Unlicensed	20 MHz/ 40 MHz	OFDM	4 X 4
IEEE 802.11ac wave	70m/ 250m	7000 Mbps	5 GHz	Unlicensed	80 MHz	64-QAM	MU-MIMO
IEEE 802.11ad	10m/ n/a	7000 Mbps	60 GHz	Unlicensed	2.16 GHz	Single Carrier/ OFDM	10 X 10
IEEE 802.11ac wave 2	70m/ 250m	7000 Mbps	5 GHz	Unlicensed	80 MHz/ 160 MHz	256-QAM	MU_MIMO 8 X 8

IoT Connectivity vs Access Networks

Indoor or Outdoor: Maximum Distance (m/km)

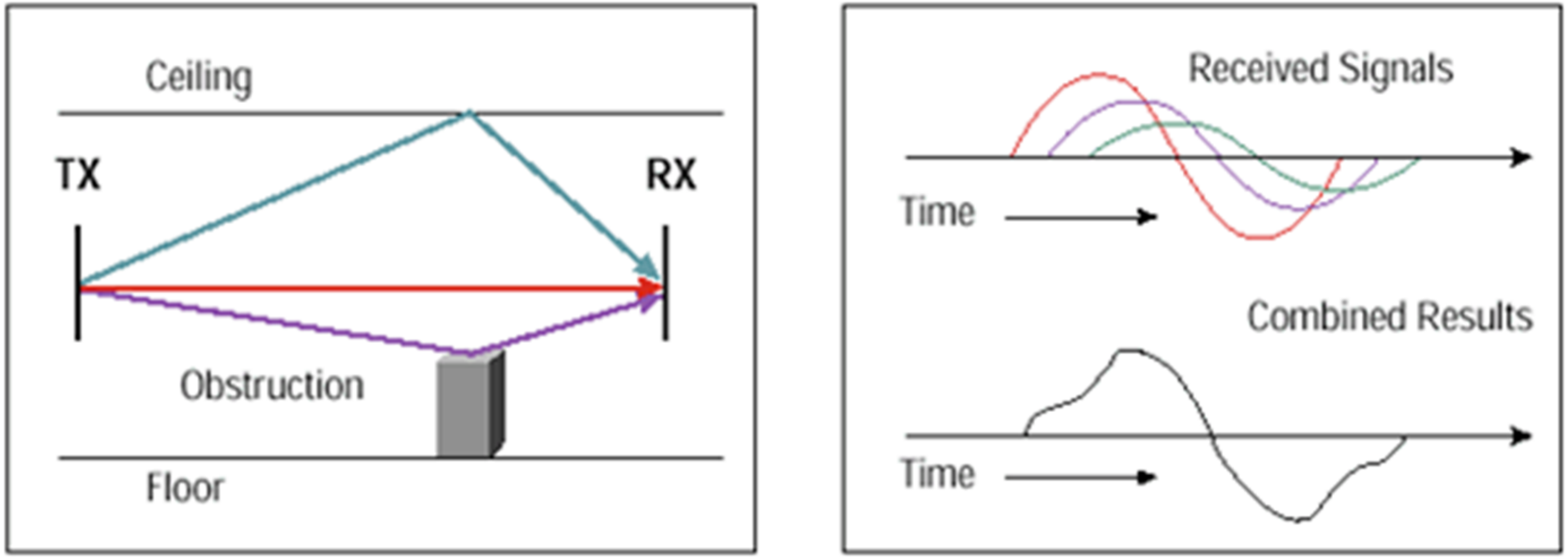




Considerations for Wireless Technologies

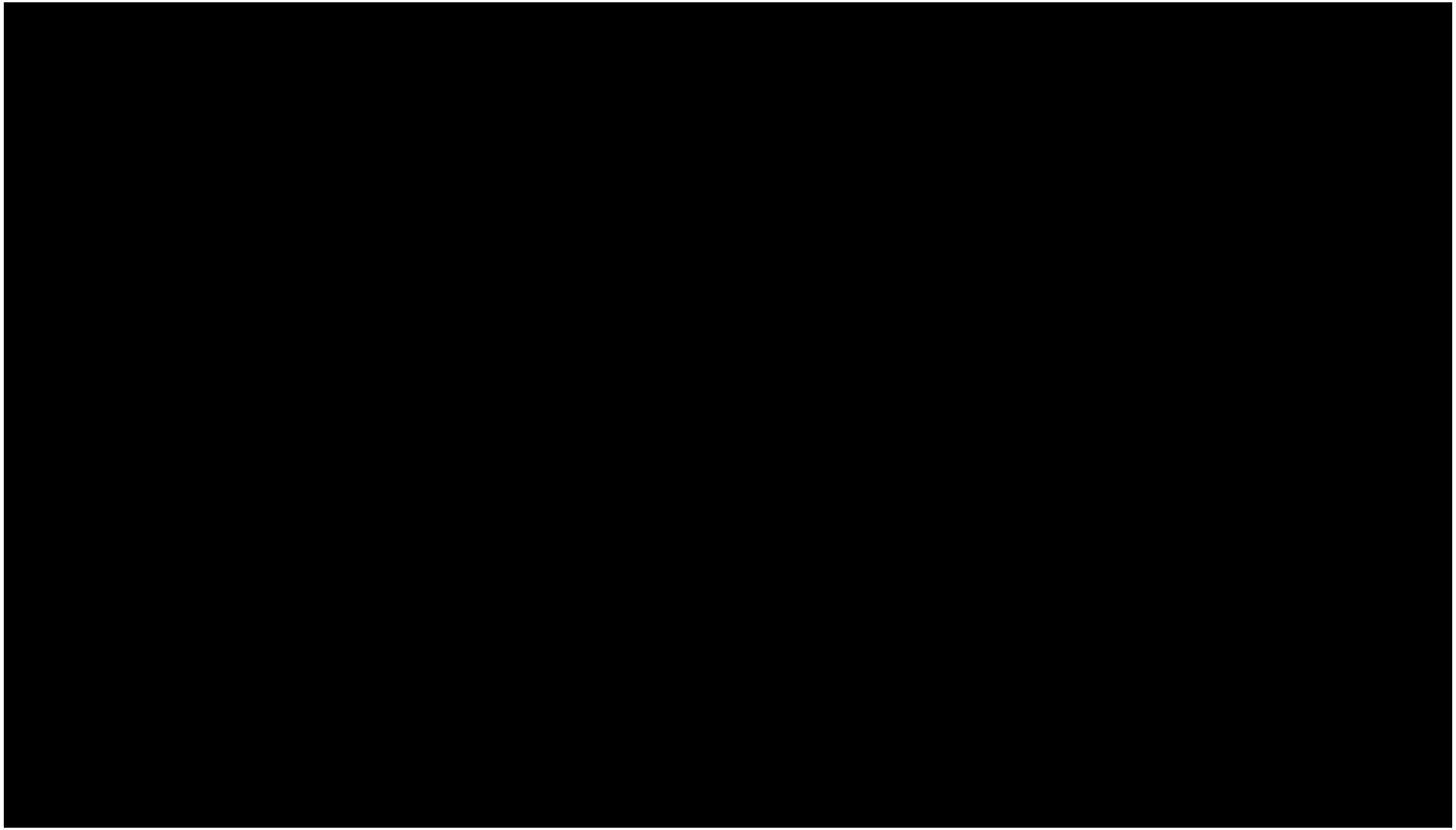
Wired / Wireless communications	A wired network uses cables to connect devices to a network so that data can be transmitted between source and destination addresses. A wireless network uses radio waves to connect devices over the air without wires.
Spectrum	Spectrum refers to the invisible radio frequencies that wireless signals travel over. Portions of electromagnetic spectrum are grouped in "bands" depending on their wavelengths–the distance over which the wave's shape repeats. Radio spectrum, we are talking about the range of radio frequencies that are used for communicating. from 30 Hertz to 300 GHz
Licensed / unlicensed spectrum	• Licensed - assigned exclusively to operators for independent usage • Unlicensed - assigned for non-exclusive usage (ie. Anyone can use) subject to some regulatory constraints, eg. restrictions in transmission power
Frequency band	A radio frequency band is a small contiguous section of the radio spectrum frequencies, in which channels are usually used or set aside for the same purpose. This is to prevent interference and allow for efficient use of the radio spectrum.
Data rate	A data rate is the theoretical maximum value that a wireless link can achieve if there were no losses or interference.
Throughput	In the real world, there will be interference and losses which will result in a lower bit rate. Throughput can be seen as a practical/realistic value that a wireless link can achieve.
Latency	Network latency is the term used to indicate any kind of delay that happens in data communication over a network. Eg. Queuing or processing times
Mobility	Mobility, or roaming, is an ability of a wireless client to maintain its association seamlessly from one access point to another securely and with as little latency as possible.
Battery/powered	Whether a sensor can be powered via battery or requires mains electrical power to operate.
Quality of Service (QoS)	Capabilities to manage the prioritization of traffic, and delay, jitter, bandwidth, and packet loss parameters on a network.
Security	Network security is any hardware or software activity designed to protect the usability and integrity of communications network performance and the data traversing it.

Signal Interferences & Multipath Signaling Transmission: attenuation, destructive



IoT Connectivity vs Access Networks

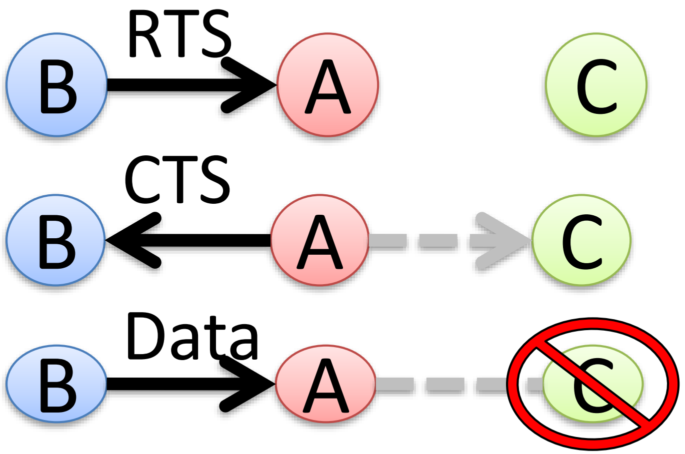
CSMA/CA cs CSMA/CD



IoT Connectivity vs Access Networks

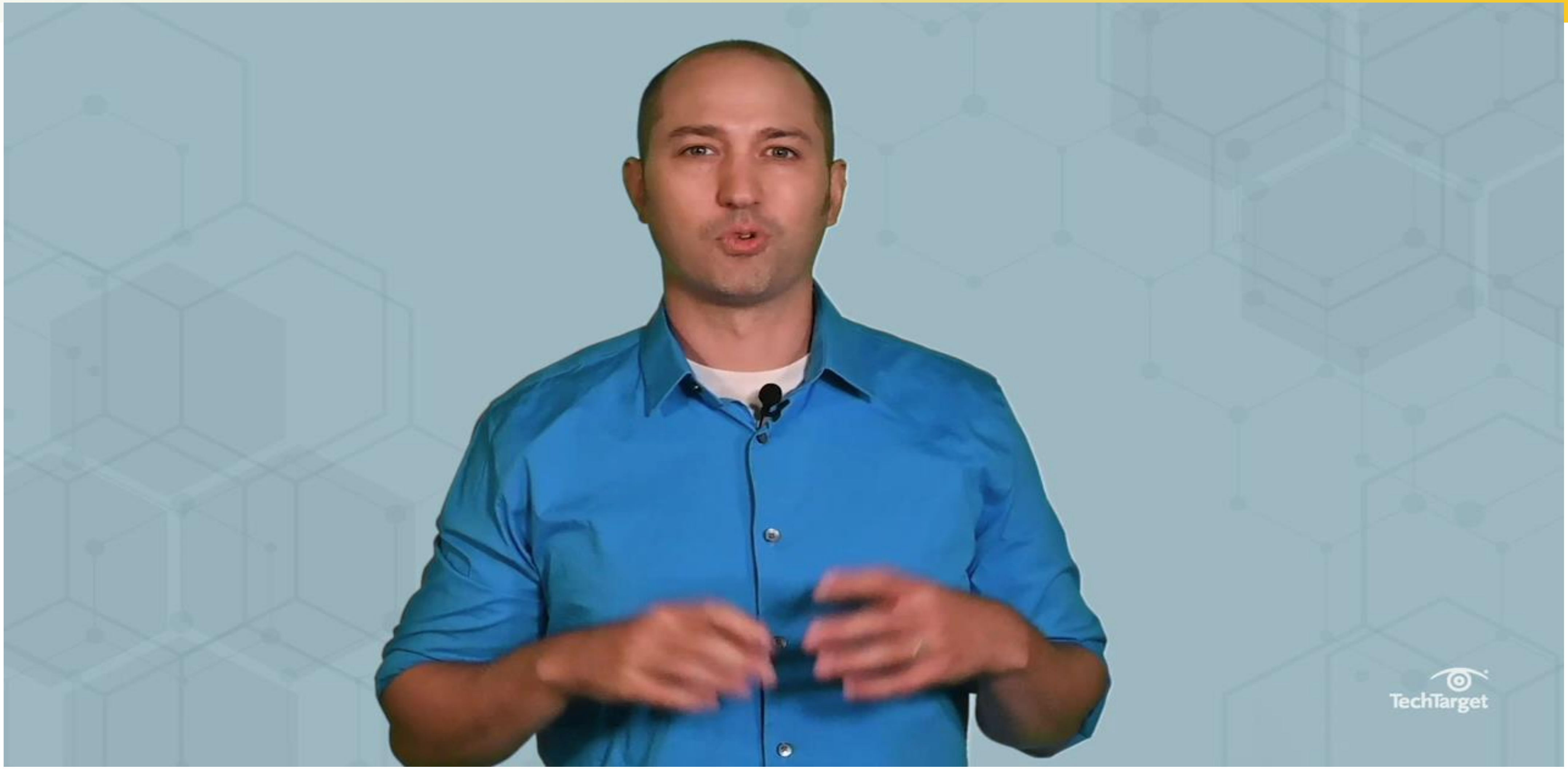
RTS/CTS

- **Idea: transmitter can check availability of channel at receiver**
- **Before every transmission**
  - Sender sends an RTS (Request-to-Send)
  - Contains length of data (in *time* units)
  - Receiver sends a CTS (Clear-to-Send)
  - Sender sends data
  - Receiver sends ACK after transmission
- **If you don't hear a CTS, assume collision**
- **If you hear a CTS for someone else, shut up**



IoT Connectivity vs Access Networks

OFDMA vs MIMO



http://youtube.com/watch?v=jU_hacMiUAc

IoT Connectivity vs Access Networks

IEEE 802.11 (Wi-Fi): High Speed Ubiquitous IoT Wireless Technology

Wi-Fi: Wireless Fidelity (Trademark)

Wifi.org (Wi-Fi Alliance as a non-profit organization)

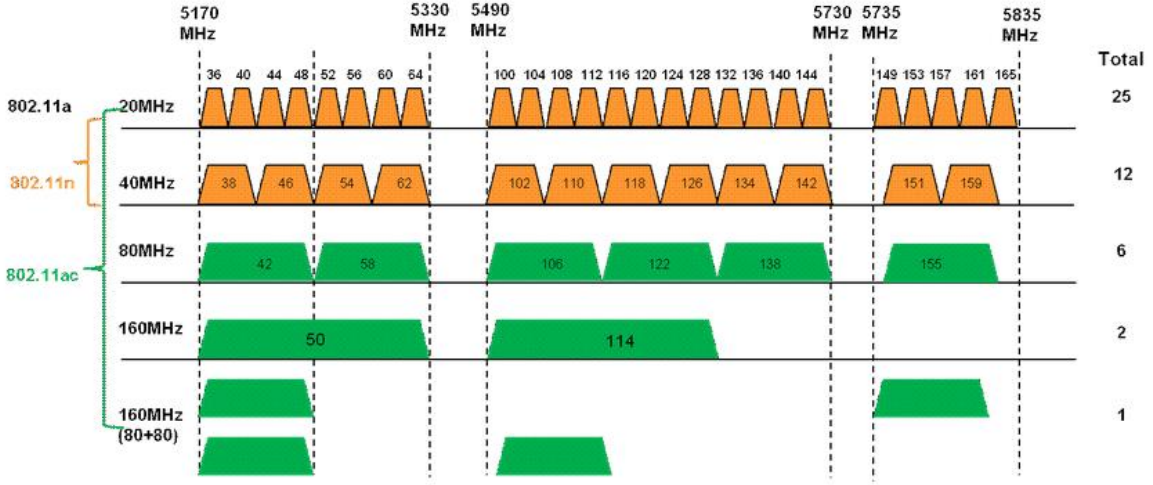
IEEE | IEEE 802.11 specifications | .11ac or Wi-Fi 5 | .11ax or Wi-Fi 6 | Wi Fi ALLIANCE

- Industrial IOT key requirement: RELIABILITY and SCALABILITY
- Wi-Fi data rate is symmetric
- All Wi-Fi versions ensure backward compatibility – in a given frequency band
- Unlicensed bands – 2.4 and 5GHz – require Access Points to comply with the local regulations, resulting in different SKUs
  - Doesn't impact Wi-Fi clients
- Next-gen under definition in IEEE 802.11be (target 2022)

Manufacturing | Transportation | Smart Cities | Oil and Gas | Roadways

IEEE 802.11: Evolution

11b	11a	11g	11n	11ac	11ax
1999	1999	2003	2009 (Wi-Fi 4)	2014 (Wi-Fi 5)	2020 (Wi-Fi 6)



Parameter	Wi-Fi 5 (802.11ac)	Wi-Fi 6 (802.11ax)
Frequency	5 GHz	2.4 and 5.0 GHz
Bandwidths (channels)	20, 40, 80+80, 160 MHz	20, 40, 80+80, 160 MHz
Access	OFDM	OFDMA
Antennas	MU-MIMO (4 × 4)	MU-MIMO (8 × 8)
Modulation	256QAM	1024QAM
Maximum data rate	3.5 Gb/s	9.6 Gb/s
Maximum users/AP	4	8

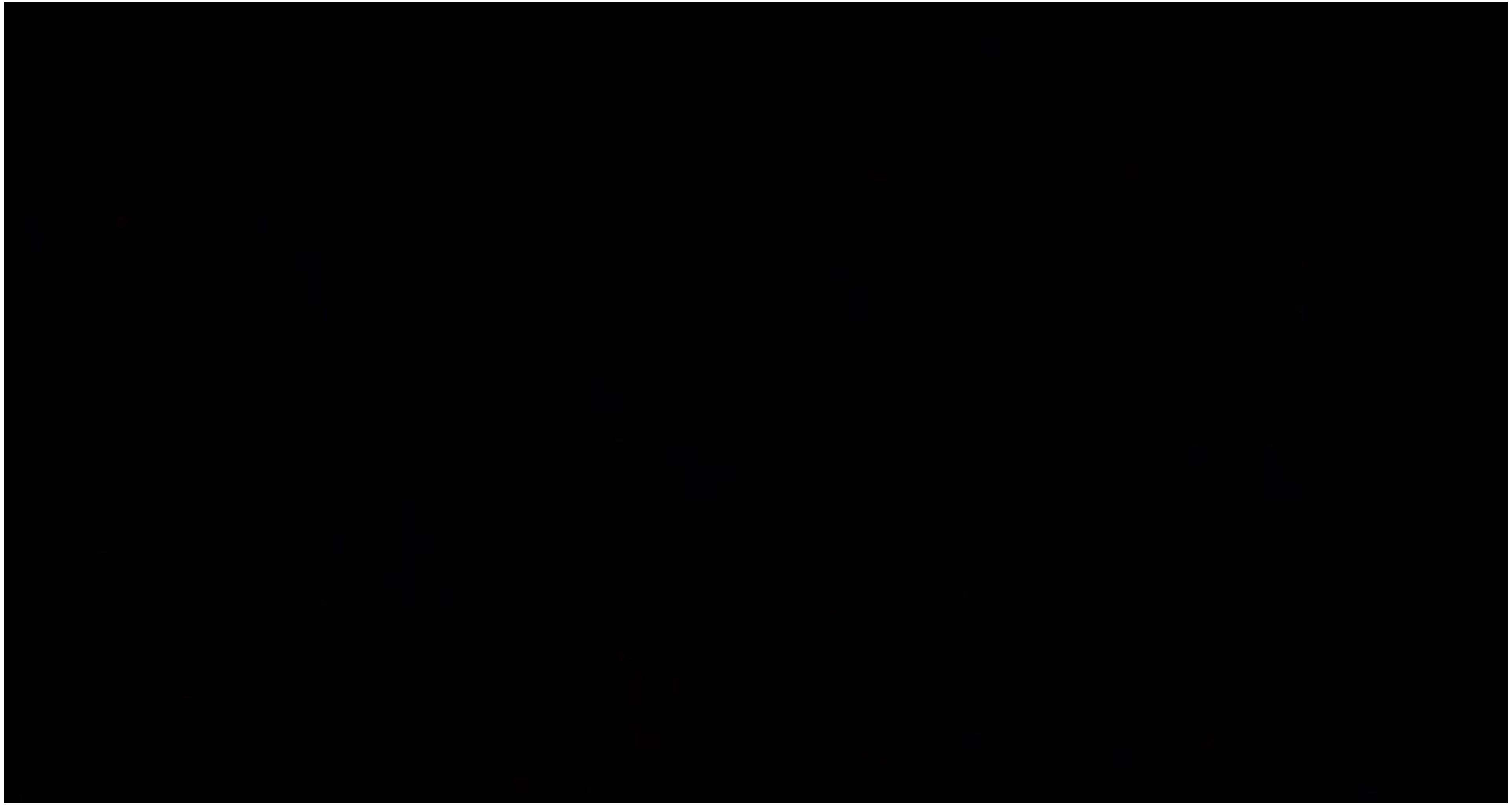
IoT Connectivity vs Access Networks

Wi-Fi vs Cell Phones

http://youtube.com/watch?v=kxLcwIMYmr0

IoT Connectivity vs Access Networks

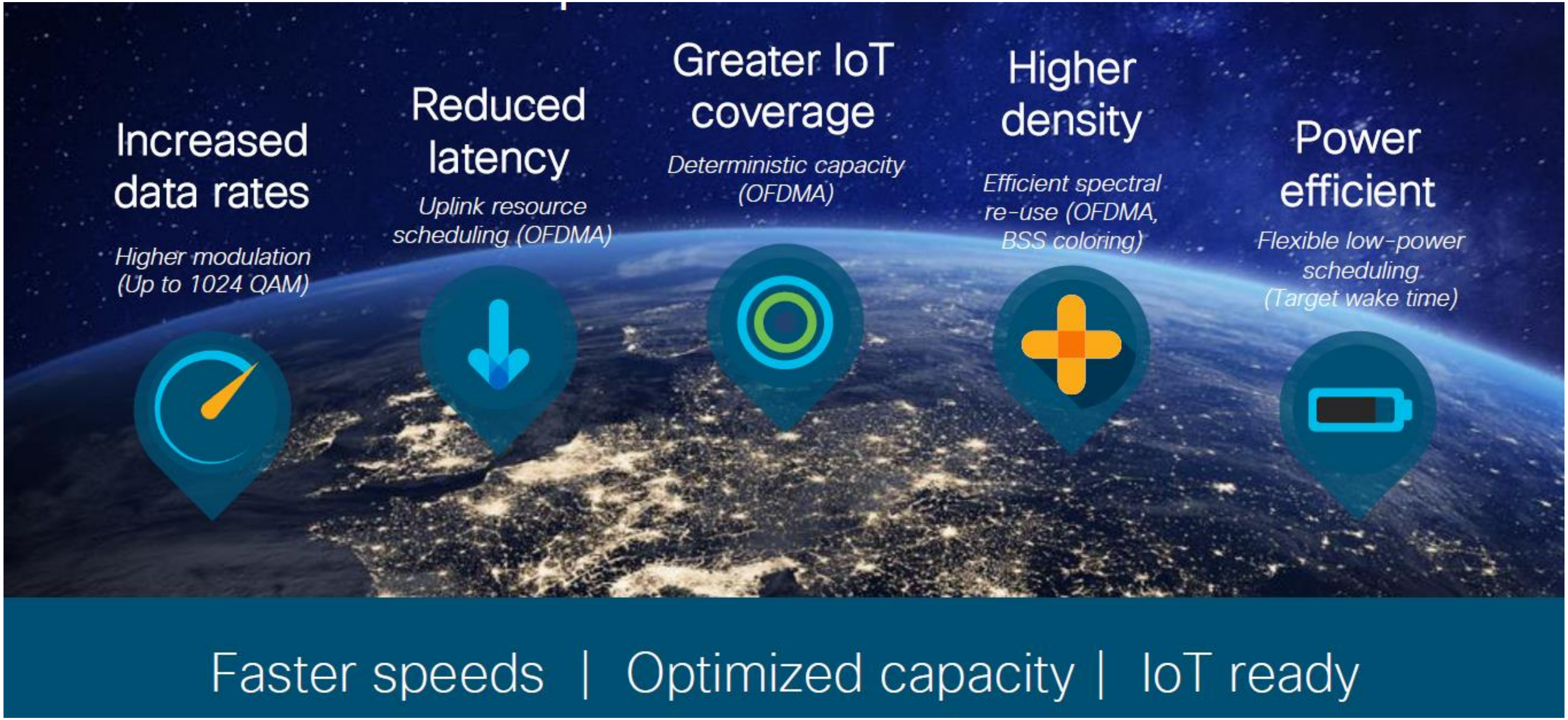
Wi-Fi 5 vs Wi-Fi 6



http://youtube.com/watch?v=kxLcwIMYmr0

IoT Connectivity vs Access Networks

Wi-Fi 6

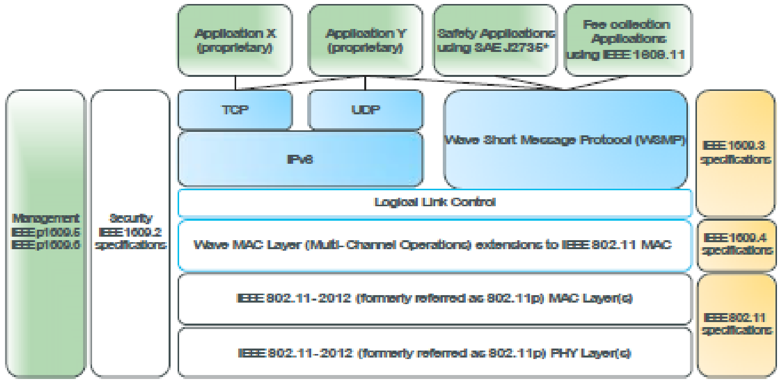


IEEE 802.11p: Dedicated Short Range Communications (DSRC)

Connected roads use cases

- DSRC is built on IEEE 802.11p extensions, running in 6GHz band
  - Data rate: 6-27 Mbps with 10 MHz Channels
  - Range up to 1000m (3000ft)
  - Will evolve with IEEE 802.11bd
- Profile is defined for IPv6 (layer-3) and IEEE 1609 WSMP (layer-2)
- Payload: less than 100 bytes for V2V, larger (+400 bytes) for I2V
- **RSU (roadside unit)** – WAVE devices that operate only when stationary and support information exchange with OBUs
- **OBU (on-board unit)** – WAVE devices that can operate when in motion and support the information exchange with RSUs or other OBUs
- Partnering

CohdaWireless



Region	Unlicensed Frequency band
North-America	5.850-5.925 GHz
Europe	5795-5815, 5855/5875-5905/5925 GHz
Japan	5770-5850 GHz
Singapore	5.855 GHz to 5.925 GHz
India	5.725 to 5.825 GHz
Australia/NZ	5,725-5,795, 5,815-5,875 MHz,
China	5,725-5,850 MHz
Korea	5,795-5,815 MHz

IoT Connectivity vs Access Networks

Cellular vs IEEE 802.11 (1/2)

	Cellular			Wi-Fi	
	4G/LTE	**Shared License Band – *CBRS Example***	**5G**	**802.11ac (Wi-Fi 5)**	**802.11ax (Wi-Fi 6)**
Positioning	Fast, IP-based, mobile broadband • Widespread/ubiquitous geographical coverage • Higher data speeds, cheaper device costs • Improved network responsiveness with lower latency and lower idle-to-active times • High spectrum efficiency + higher network capacity • Backwards compatibility • Interoperability through all IP network • Enhancements to security and Quality of Service • Mobility support • Country/SP dependent	Citizen broadband radio service • Multi-technology (including WiFi) support capable • Create own private LTE networks, replacing/ supplementing Wi-Fi • Efficient use of wireless spectrum - less interference • Ecosystem dependent • Neutral host feature allows connectivity to multiple LTE providers to better support dense environments • Potential benefits for multi-IoT application deployments • US only, *Netherlands, UK, Germany, Sweden, Japan planning something similar*	Enhanced mobile broadband, and wireless for industry • Higher data rates and improved quality of experience for wireless broadband • More device capacity for IoT and enterprise deployments • Support for new high bandwidth and low latency services • Improved performance with less spectrum interference • Virtualization capabilities • Connections in hard to reach areas and wired replacement • Mobility support • Country/SP dependent • High b/w backhaul links	5th generation wireless • Current generation of wireless with speed performance comparable to standard wired connections • Higher speeds over wider bandwidths than previous standards • More reliable long-range transmissions over previous standards • Improved performance in environments with obstructions • Backwards compatibility • Improved multi-user performance • High b/w backhaul links	6th generation wireless • 40% higher data rates • More predictable performance in dense client environments • Simultaneous comms with multiple clients • Real time applications, up to 75% less latency • Increased capacity + range • Improved power efficiency and battery of devices • More robust outdoor performance • Easier to guarantee near wall-to-wall indoor coverage vs 5G • Backwards compatibility • High b/w backhaul links
Features	• Indoor/outdoor • Max data rate 300Mb-1Gb • Av. user speed 15-50Mb • 10,000 connections per Km2 • 30-50ms latency • QoS, predictability • Security • Long range • Asymmetric U/D bandwidth	• Indoor/outdoor • Low latency and improved scale of connections vs Wi-Fi • LTE services 1Gb indoors - 5-10x outdoor line-of-sight • mobility/roaming across access points • More deterministic performance for real-time • Asymmetric U/D bandwidth	• Indoor/outdoor • Target 10Gb UL,20Gb DL • Target user 50Mb UL, 100Mb DL • 1 million devices per km2 • Network slicing • Sub-1 millisecond latency • Predictability and QoS • Long range • Asymmetric U/D bandwidth	• Indoor/outdoor • Max data rate 6.93Gb • Multi-user access • Medium range • Variable latency • Symmetric U/D bandwidth	• Max data rate 9.6Gb • Medium range • Multi-user access • More deterministic behaviour • Improved indoor + outdoor coverage • Reduced latency up to 75% • Split network capacity among groups of devices • Symmetric U/D bandwidth

IoT Connectivity vs Access Networks

Cellular vs IEEE 802.11 (2/2)

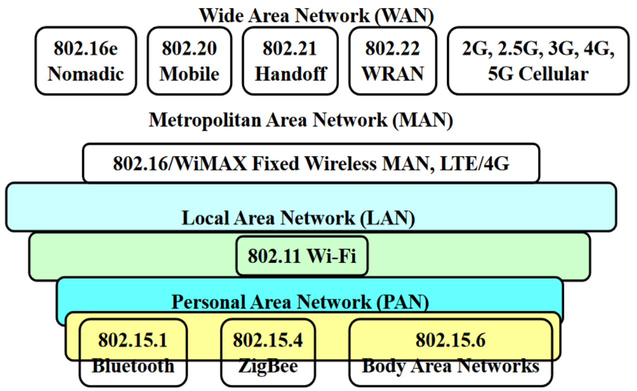
	Cellular			Wi-Fi	
	4G/LTE	Shared License Band – *CBRS Example*	5G	802.11ac (Wi-Fi 5)	802.11ax (Wi-Fi 6)
Technical attributes	• Channel bandwidths 1.4, 3, 5, 10, 15, 20Mhz • 300km/h mobility • Licensed and unlicensed	• Spectrum: 3.55 to 3.7GHz • Licensed • Time Division Duplex (TDD) • Does not have to be line of sight	• Spectrum: Low-band: 600M-900MHz / Mid-band: 2.5G - 4.2GHz / Millimeter wave (high-band): 24GHz - 47GHz • Licensed • Seamless mobility – up to 500km/h mobility • Network slicing • Dynamic traffic optimization • Dynamic policy control • Function/resource elasticity • Granular QoS/prioritization • Multiple-access connectivity	• Spectrum: 5GHz • Unlicensed • Up to 8 spatial streams • Standards based beam forming • MU-MIMO • Stronger Clear-Channel Assessment (CCA) requirements • RTS/CTS with bandwidth indication	• Spectrum: 2.4GHz, 5GHz (Future 1 and 6 GHz) • Unlicensed • OFDMA scheduling to reduce overhead and latency • MU-MIMO • MAC special reuse with colour codes • Denser modulation
Example Use Cases	• Smart metering • Distribution automation • Smart agriculture • Renewable/variable energy • Smart lighting • Mobile connectivity services	• Last mile and line of site connectivity • Mass IoT sensor deployments • Multi IoT application deployments • Private LTE networks • Private wireless • Neutral host networks	• High performance media applications • Smart vehicle-to-X transportation/autonomous • Dense outdoor sensor and IoT device connectivity • Last mile and line of site connectivity • Location based services • Virtual + augmented reality • Smart metering • Distribution automation • Renewable/variable energy • Smart agriculture • Edge compute	• Pervasive indoor and medium range outdoor connectivity • User Mobility • Location based services • Sensor connectivity • Smart machines – monitoring • Virtual + augmented reality • Smart retail + customer experience	• Pervasive indoor and medium range outdoor connectivity • User Mobility • Location based services • Sensor connectivity • Smart machines – monitoring and control • Virtual + augmented reality • Smart retail + customer experience • Time sensitive applications

IEEE 802.15 Projects: Personal Area Network

- IEEE 802.15.1-2005: Bluetooth1.2
- IEEE 802.15.2-2003: Coexistence Recommended Practice
- IEEE 802.15.3-2016: High Rate (55 Mbps) Multimedia WPAN, includes 3c-2009 mm wave PHY, 3b-2005 High rate WPAN
- IEEE 802.15.3d-2017: 100 Gbps point-to-point PHY
- IEEE 802.15.3e-2017: High-Rate close proximity point-to-point MAC and PHY
- IEEE 802.15.3f-2017: High-rate wireless multi-media networks using mm waves
- IEEE 802.15.4a-2007: Precision Ranging
- IEEE 802.15.4c-2009: Chinese 314-316, 430-434, 779-787 MHz
- IEEE 802.15.4d-2009: Japanese 950 MHz
- IEEE 802.15.4e-2012: MAC Enhancements
- IEEE 802.15.4f-2012: PHY for Active RFID
- IEEE 802.15.4g-2012: PHY for Smart Utility Networks
- IEEE 802.15.4j-2013: Medical Body Area Network 2.36-2.4 GHz
- IEEE 802.15.4k-2013: Low Energy Critical Infrastructure Monitoring PHY
- IEEE 802.15.4m-2014: TV White Spaces PHY –between 56 MHz and 862 MHz
- IEEE 802.15.4n-2016: China Medical Band PHY
- IEEE 802.15.4p-2014: Rail (Train) Communications & Control PHY
- **IEEE 802.15.4-2015: Low Rate (250kbps) WPAN –ZigBee**
- IEEE 802.15.4md:Maintenance of IEEE 802.15.4-2015
- IEEE 802.15.4q-2016: Ultra Low Power PHY
- IEEE 802.15.4s-2018: System resource management capability
- IEEE 802.15.4t-2017: High rate (2 Mbps) PHY
- IEEE 802.15.4u-2016: 865-867 MHz band in India
- IEEE 802.15.4v-2017: Enabling use of regional sub-GHz bands (4n ,4q, 4t, 4u)
- IEEE P802.15.4w: Low-Rate Low-Power Wide Area Network (LPWAN) extension to 802.15.4 PHY to cover 10-15 km
- IEEE P802.15.4x: Field Area Network extensions for devices with no battery or very limited battery consumption (Smart Utility Network)
- IEEE P802.15.4y: Security next generation using AES-256
- IEEE P802.15.4z: Enhanced impulse radio Ultra-Wide Band(UWB)
- IEEE 802.15.5-2009: Mesh Networking. Full/partial meshes. Range Extension
- IEEE 802.15.6-2012: Body Area Networking. Medical and entertainment. Low power
- IEEE 802.15.7-2011: Short Range Optical Wireless
- IEEE P802.15.7r1: Optical wireless (infrared, ultraviolet, visible light)
- IEEE P802.15.7m: Maintenance of 802.15.7-2011
- IEEE 802.15.8-2017: Peer Aware Communications
- IEEE 802.15.9-2016: Key Management Support
- IEEE 802.15.10-2017: Routing packets in dynamically changing wireless networks
- IEEE P802.15.10a: Routing mode additions. Automated discovery of nodes and route configuration
- IEEE P802.15.12: Upper Layer Interface (ULI) to harmonize fragmentation, configuration etc. for all 802.15.4 (Upper L2 and interface to L3)
- IEEE P802.15.13: Multi-Gigabit/s Optical Wireless with ranges up to 200m
- IEEE 802.15 IG6T: Consolidate Link Layer Control interest group
- IEEE 802.15 IGdep: Enhanced Dependability interest group
- IEEE 802.15 IGvat: Vehicular Assistive Technology

- IEEE P802.15.7m: Maintenance of 802.15.7-2011
- IEEE 802.15.8-2017: Peer Aware Communications
- IEEE 802.15.9-2016: Key Management Support
- IEEE 802.15.10-2017: Routing packets in dynamically changing wireless networks
- IEEE P802.15.10a: Routing mode additions. Automated discovery of nodes and route configuration
- IEEE P802.15.12: Upper Layer Interface (ULI) to harmonize fragmentation, configuration etc. for all 802.15.4 (Upper L2 and interface to L3)
- IEEE P802.15.13: Multi-Gigabit/s Optical Wireless with ranges up to 200m
- IEEE 802.15 IG6T: Consolidate Link Layer Control interest group
- IEEE 802.15 IGdep: Enhanced Dependability interest group
- IEEE 802.15 IGvat: Vehicular Assistive Technology

- IEEE 802.15 IGguide: Guide for 15.4 use interest group
- IEEE 802.15 IGhrrc: High Rate Rail Communications interest group
- IEEE 802.15 IGTHz: Terahertz interest group
- IEEE 802.15 SCwng: Wireless Next-Generation standing committee
- IEEE 802.15 SCmaint: Maintenance standing committee
- IEEE 802.15 SCietf: IETF Liaison



connectivity standards alliance

https://csa-iot.org/

IoT Connectivity vs Access Networks

IEEE 802.15.1: Bluetooth

IEEE 802.15.4: Design Challenges

- **Battery powered**:
  Maximize battery life.
  A few hours to a few years on a coin cell.
- **Dynamic topologies**:
  Short duration connections and then device is turned off or goes to sleep
- **No infrastructure**
- **Avoid Interference:**
  due to larger powered LAN devices
- **Simple and Extreme Interoperability**:
  Billions of devices. More variety than LAN or MAN
- **Low-cost**:
  A few dollars

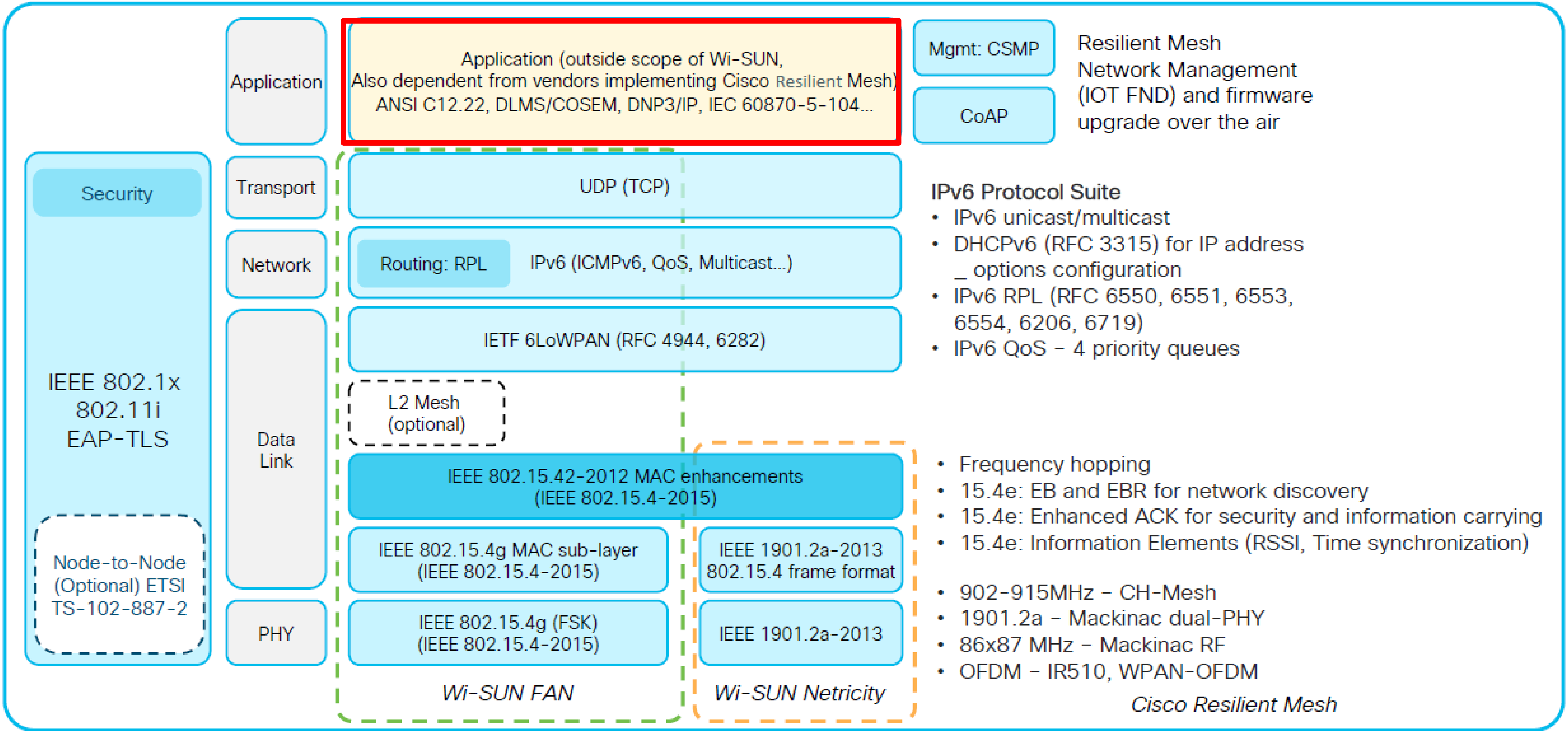
IoT Connectivity vs Access Networks

IEEE 802.15.4: Personal Area Network

	IEEE 802.15.4-2006	IEEE 802.15.4g/e SUN PHYs	Comments
Frequency bands	868 MHz 1-3 channels, 902-928 MHz 10-30 channels 2450 MHz 16 channels	In addition of 802.15.4-2011 frequency bands 169, 450-470, 470-510, 863-870, 1427-1518 MHz	• Frequency bands availability per region/country • # channels: 802.15.4-2003 - 802.15.4-2006 • 802.15-4-2011: 314-316, 430-434 and 779-787 MHz bands for China
Modulation	BPSK, ASK (Sub-GHz) O-QPSK (2.4GHz)	MR-FSK, MR-OFDM and MR-O-QPSK	BPSK/O-QPSK in 802.15.4-2003 802.15.4-2011 adds modulations 802.15.4g add 3 new PHY SUN modulation
Max. theoretical Data Rate	Up to 20, 40 and 250 kb/s	Up to 1200 kb/s (OFDM)	Frequency band and modulation dependent
Maximum PSDU size	127 bytes	2047 bytes	Better aligned with IPv6 MTU (1280 bytes)
FCS	16 bits	32 bits	Better error protection
Information Elements	No	Yes, 15.4e	Allow vendor specific information
PAN ID	0-65534	0-65534	Identifies a WPAN
MAC Address	16 bits or 64 bits	16 bits or 64 bits	16 bits = locally managed, 64 bits = EUI-64
Usage	Zigbee, WirelessHart, ISA100	Wi-SUN	

IoT Connectivity vs Access Networks

Wi-SUN (Smart Ubiquitous Networks)



WirelessHart vs ISA 100.11a

	WirelessHart	ISA100.11a
Frequency bands	IEEE 802.15.4-2006 2.4GHz, 16 channels	IEEE 802.15.4-2006 2.4GHz, 16 channels
Data Rate	250kbs	250kbs
Standard	IEC 62591	IEC 62734
Topology	TDMA/CSMA based wireless mesh	TDMA/CSMA star, mesh, star-mesh topologies
Channel hopping	fixed channel hopping table 10 msec time slot	multiple channel hopping tables variable slot time, default 10 msec
	Based on HART addressing	6LoWPAN, IPv6 and UDP
Vendors	Emerson, ABB, Siemens, Endress+Hauser	Honeywell, Yokogawa, GE
Specifications	https://fieldcommgroup.org/hart-specifications	https://www.isa.org/store/products/product-detail/?productId=118261

I552SA/SD

3 Antennas (2.4/5 GHz)
AC and DC Power models
Integrated ISA100 and WirelessHART radio

I552SA/SD

6 Antennas (3x2.4/3x5 GHz)
DC Power
WirelessHART Gateway

IW6300

Mesh architecture support based on 802.11 AC Wave 2

WirelessHart and ISA100 as add-on module from partners

IoT Connectivity vs Access Networks

Wi-SUN

IEEE 802.15.4: Personal Area Network

Overview

- Low Rate Wireless Personal Area Network (LR-WPAN)
- 2.4 GHz (most common). 16 x 5-MHz channels
- 250 kbps PHY ⇒50 kbps application data rate
- Peak current depends upon symbol rate ⇒multilevel 4b/symbol)
- Similar to 802.11: Direct Sequence Spread Spectrum, CSMA/CA, Backoff, Beacon, Coordinator (similar to Access point)
- Lower rate, short distance ⇒Lower power ⇒Low energy
- Each node has a 64-bit Extended Unique ID (EUI-64):

U/M	G/L	OUI	40 bits assigned by the manufacturer
1b	1b	22b	40b

- No segmentation/reassembly.
  Max MAC frame size is 127 bytes with a payload of 77+ bytes.

IEEE 802.15.4: Personal Area Network

Topologies

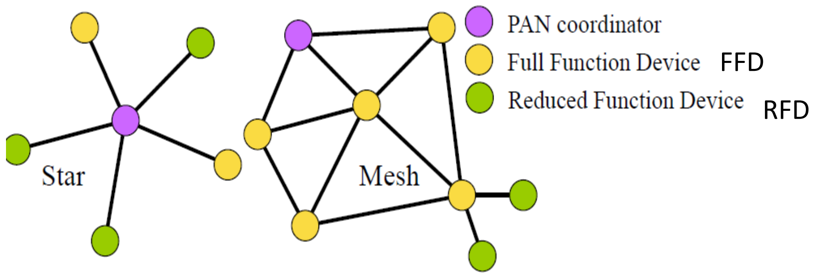
- FFDs can become coordinator and can also route messages to other nodes
- RFDs cannot become coordinator and can only be a leaf
- FFD that starts a PAN becomes the coordinator
- In star topology, all communication is to/from the coordinator
- In P2P topology, FFDs can communicate directly also.
- Each piconet has a PAN ID and is called a cluster.
- Nodes join a cluster by sending association request to the coordinator.

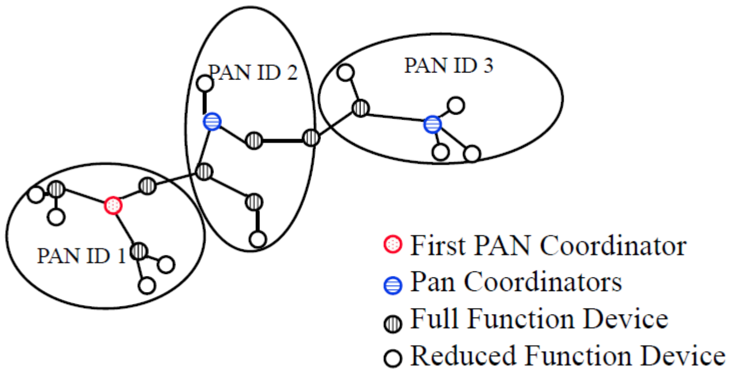
  Coordinator assigns a 16-bit short address to the device.

  Devices can use either the short address or EUI-64 address.

- A coordinator can ask another FFD to become a coordinator for a subset of nodes.

  Tree ⇒No loops

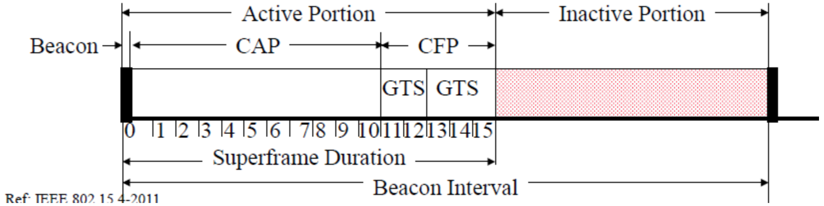


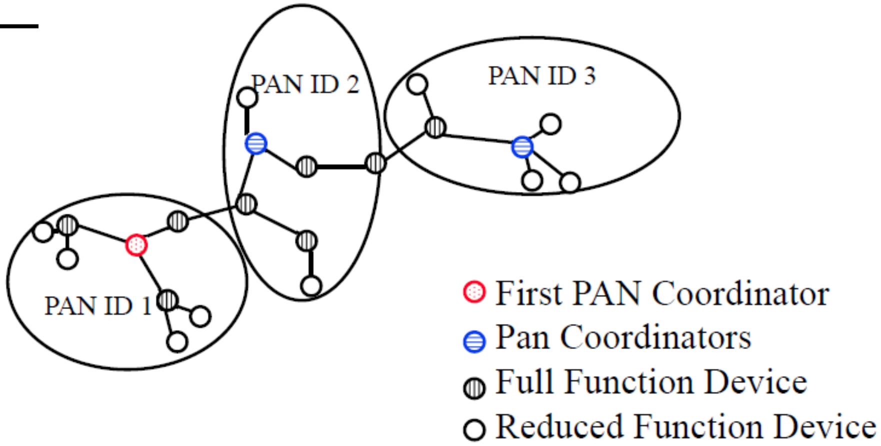


MAC

Beacon-Enabled CSMA/CA

- Coordinator sends out beacons periodically
- Part of the beacon interval is inactive ⇒Everyone sleeps
- Active interval consists of 16 slots
- Guaranteed Timed Slots (GTS):
  For real-time services. Periodic reserved slots.
- Contention Access Period (CAP). Slotted CSMA.

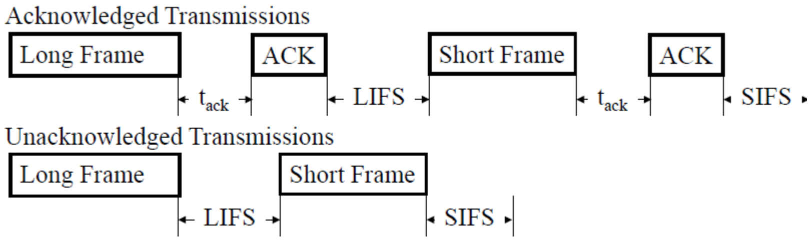




- **Beaconless Operation:** Unslotted CSMA
  If coordinator does not send beacons, there are no slots
- **Acknowledgements if requested by the sender.**
- Short inter-frame spacing (SIFS) if previous transmission is shorter than a specified duration.
  Otherwise, Long inter-frame spacing (LIFS)

802.15.4 CSMA/CA

- Wait until the channel is free.
- Wait a random back-off period If the channel is still free, transmit.
- If the channel is busy, backoff again.
- Backoff exponent limited to 0-2 in battery life-extension mode.
- Acknowledgement and Beacons are sent without CSMA-CA.

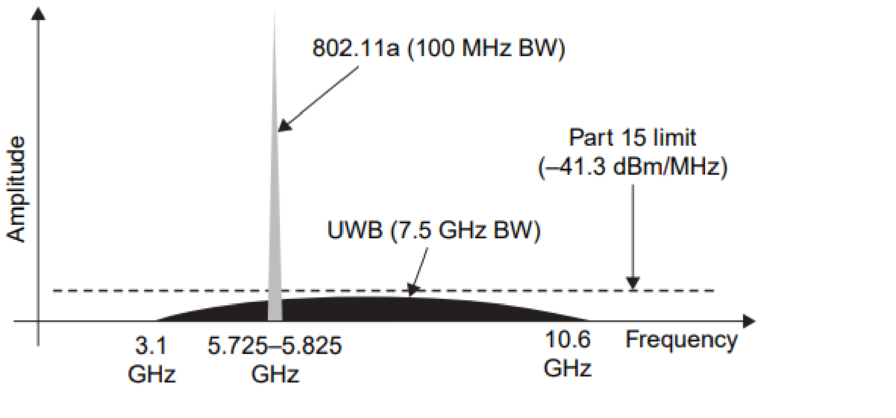


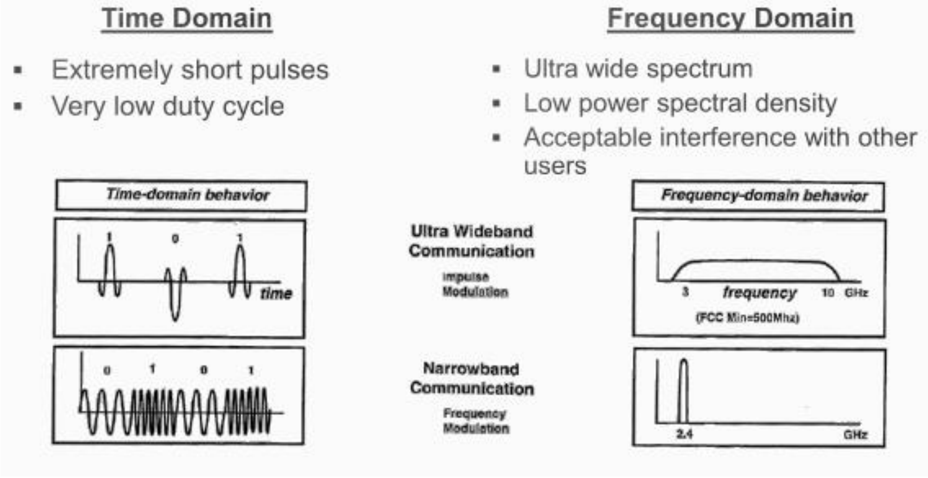
Ultra-Wideband (UWB) 1/2

- An impulse in time domain results in a ultra wide spectrum in frequency domain and essentially looks like a white noise to other devices.
- FCC rules restrict the maximum noise generated by a wireless equipment (0 dBm = 1mW, -40 dBm = 0.1 μW)
- It is possible to generate very short (sub-nano sec) pulses that have spectrum below the allowed noise level

  ⇒Possible to get Gbps using 10 GHz spectrum
- FCC approved UWB operation in 2002
- UWB can be used for high-speed over short distances
- UWB can see through trees and underground (radar)

  ⇒collision avoidance sensors, through-wall motion detection
- Position tracking: cm accuracies. Track high-value assets
- Sub-nanosecond impulses are sent many million times per second
- Became feasible with high-speed switching semiconductor devices
- Pulse width = 25 to 400 ps
- Impulses may be position, amplitude, or polarity modulated
- 0.25 ns Impulse ⇒4 B pulses/sec ⇒100's Mbps
- 802.15.4 uses pulse position and binary phase shift keying modulation





Ultra-Wideband (UWB) 2/2

Advantages of UWB

- Very low energy consumption: Good Watts/Mbps
- Line of sight not required. Passes through walls.
- Sub-centimeter resolution allows precise motion detection
- Pulse width much smaller than path delay ⇒Easy to resolve multipath⇒Can use multipath to advantage
- Difficult to intercept (interfere)
- All digital logic ⇒Low cost chips
- Small size: 4.5 mm^2 in 90 nm process for high data rate designs

Direct Sequence (DS-UWB)

- Championed by Motorola/XtremeSpectrum
- Uses CDMA with multiple chips per bit
- Chips are encoded using pulse
- This is the scheme used in 802.15.4
- Low power density ⇒Good for body area network

IEEE 802.15.4e Enhancements

- Low latency deterministic operation: pre-assigned slots
- Channel adaptation: Different channels used by different nodes for contention free period
- Time slotted channel hopping: Higher layers coordinate the slot allocation along with its frequency. Good for harsh industrial environments.
- Each device can select its listening channel
- Transmitter and receiver coordinate their cycles (very low duty cycle)
- Transmit only when requested by receiver

Summary

- IoT fueled initially by smart grid is resulting in several competing protocols: Bluetooth Smart, ZigBee Smart, ...
- IEEE 802.15.4 is a low-data rate wireless personal area network and is the PHY and MAC layer used by many IoT protocols, such as ZigBee, and WirelessHART.
- 802.15.4 uses full function and reduced function devices. FFDs can act as coordinator. Allows a star, mesh, or a cluster tree topology.
- Uses Slotted/Unslotted CSMA/CA. Supports Guaranteed timed slots for low-latency application.
- UWB allows transmission with very low average power spread over a large band.

NB-IoT vs LoRaWAN vs IEEE 802.15.4 (1/2)

IoT Focused

	NB-IoT	LoRaWAN	802.15.4	LTE-M
Positioning	**Narrowband IoT** • High density of devices • Extremely small power budgets of devices • Very long range • May be able to provide deeper building and underground penetration than similar technology • Challenges with firmware-over-the-air (FOTA) or large file transfers • Can co-exist with 2G, 3G, and 4G mobile networks. • Same security benefits as mobile networks • No mobile use cases • Higher power consumption than LoRaWAN	**Long Range (LoRa)** • Designed for sensors and applications that need to send small amounts of data over long distances a few times per hour from varying environments • Single GW covers wide area • Multi-year battery lifetime/low power consumption • Communications robustness • 2 layers of security • Uses device classes to optimize a variety of end application profiles • Interoperability between manufacturers	**Wireless Sensor Networks** • aimed at providing the essential lower network layers for a wireless personal area network, WPAN • low-cost, low-speed ubiquitous communication between devices • Provides foundation for a variety of different higher layer standards such as Zigbee, Wireless HART, ISA100, 6LowPAN (table in notes section)	**Long-Term Evolution for Machines** • Compatible with existing LTE network • supports real-time device communication • Device cost and power consumption reduction • Extended battery life through sleep and power saving modes • Lower service cost vs LTE due to reduced data b/w • Mobile use cases
Features	• 20-60kb/s • Long range • Less latency and higher throughput than LoRa • Network security mechanisms • Asymmetric U/D bandwidth	• Unlicensed band • Indoor/outdoor • 250 bps to 50 kbps • Long range • Security mechanisms • Symmetric/Asymmetric U/D bandwidth dependent on RF region • Multicast support • Firmware Over the Air FOTA	• Unlicensed band, i.e. EU868, US915 • Private infrastructure • 50kbs – 1.2Mbs (US915) • 50kbs – 300kbs (EU868) • Medium range • Simple or no QoS requirements • Security mechanisms	• 1Mb up / 384kb down • High Latency Communication • Indoor/outdoor • Extended range • Asymmetric U/D bandwidth

NB-IoT vs LoRaWAN vs IEEE 802.15.4 (2/2)

IoT Focused

	NB-IoT	LoRaWAN	802.15.4	LTE-M
Technical attributes	• 200-kHz GSM spectrum • In-band and Guard-band LTE, standalone GSM • Half-duplex	• Spectrum variable by region eg. 868 - 868.6 Europe, 902-928 USA • Frequency hopping • Tunable variable data rates • Network Session Key shared between end-device and network server • Application Session Key (AppSKey) shared end-to-end at the application level • AES algorithms are used to provide authentication and integrity of packets to the network server and end-to-end encryption to the application server.	• Spectrum variable by region, eg. 868 - 868.6 Europe, 902-928 USA, 2.4Ghz global • PHY and MAC layers only • Selectable levels of security – privacy (encryption), sender authentication, message integrity • Flexible protocol design suitable for many applications • handshake protocol for transfer reliability	• Spectrum: 1.4 MHz bandwidth • Battery life of devices extended through sleep mode or power saving mode (PSM) and through extended discontinuous reception (LTE eDRX) • High Latency Communication o Support for extended coverage o LTE-M Half Duplex Mode/Full Duplex o Support of Category M1 device o VoLTE support o SMS
Use Cases	• Connected white goods • Connected animals • Logistics • Smart metering • Street lighting • Smart parking • Asset tracking • Industrial sensor monitoring	• Smart agriculture • Industrial sensor reading • Medical sensor reading • Smart cities • Environmental sensors • Smart metering • Logistics • Home automation	• General automation • Home automation • Smart Grids • Industrial sensor networks • Asset location	• Healthcare devices • Voice (if supported by carrier) • Smart city sensors • Environmental sensors • Emergency data • Precision tracking • Wearables • Telematics

Industrial IoT Wireless Technologies [CISCO]

	LoRaWAN	Resilient Mesh	WiFi	4G/LTE – 5G
Topology	Point to multipoint	Mesh Point to Multipoint (Leaf mode)	Point to Multipoint Mesh	Point to multipoint
Coverage Range (Radio signal is the real value)	~ 2k-10km	~1.5km per hop up to 8 hops	~100m (300 feet)	~ 2k-10km (cell dependent
Data Rate	250bs-21kbs	50kbps -1.2Mbps	11Mbs (.b) 1.7 Gbs (.ac W2) 9.6 Gbs (.ax)	27kbs(DL)/65kbs(UL) NB-IOT HDx 300kbs/375kbs LTE Cat M1 300 Mbps (DL)/50Mbs (UL) LTE Cat.6 to 5G (500Mbs UL/5Gbs DL) on today's modem
Public SP vs Private Networks	Private/Public SP	Private	Private/Public SP	Public SP/Private (i.e. US CBRS)
Batteries powered devices	Optimized lifetime (+10 years)	Not in FAN 1.0	Limited lifetime (months)	NB-IOT provides good lifetime
Eco-system (endpoints)	*****	*	*****	****
TCO	Low	Low-Medium	Medium-High	Medium-High

References

[VB17] Ovidiu Vermesan and Joel Bacquet, “**Cognitive Hyperconnected Digital Transformation: Internet of Things Intelligence Evolution**”, River Publishers, 2017.

[BHC+18] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, “**The Industrial Internet of Things (IIoT): An Analysis Framework**”, Computers in Industry 101, October 2018.

[BS19] Rajkumar Buyya and Satish Narayana Srirama, “**Fog and Edge Computing: Principles and Paradigms**”, Wiley Series on Parallel and Distributed Computing, 2019.

[RS19] Ammar Rayes and Samer Salam, “**Internet of Things from Hype to Reality: The Road to Digitization**”, 2nd Edition, Springer, 2019.

[LEA20] Perry Lea, “**IoT and Edge Computing for Architects: Implementing edge and IoT systems from sensors to clouds with communication systems, analytics, and security”**, 2nd Edition, Packt Publishing, 2020.

[GG20] Patrick Grossetete and Stephen Goodman, “**Wireless Technologies and Use Cases in Industrial IoT**”, BKKIOT-1775, Cisco Live, January 2020.

Lecture Course: Wireless and Mobile Networking

https://www.cse.wustl.edu/~jain/cse574-20/index.html

https://learning.oreilly.com/videos/internet-of-things/9780137592135/9780137592135-iott_00_00_00_00/

ITCS447
Lecture 9
IoT Communication and Networking

Part 2: IoT Application Protocols

Asst. Prof. Dr. Thitinan Tantidham

มหาวิทยาลัยมหิดล
Mahidol University

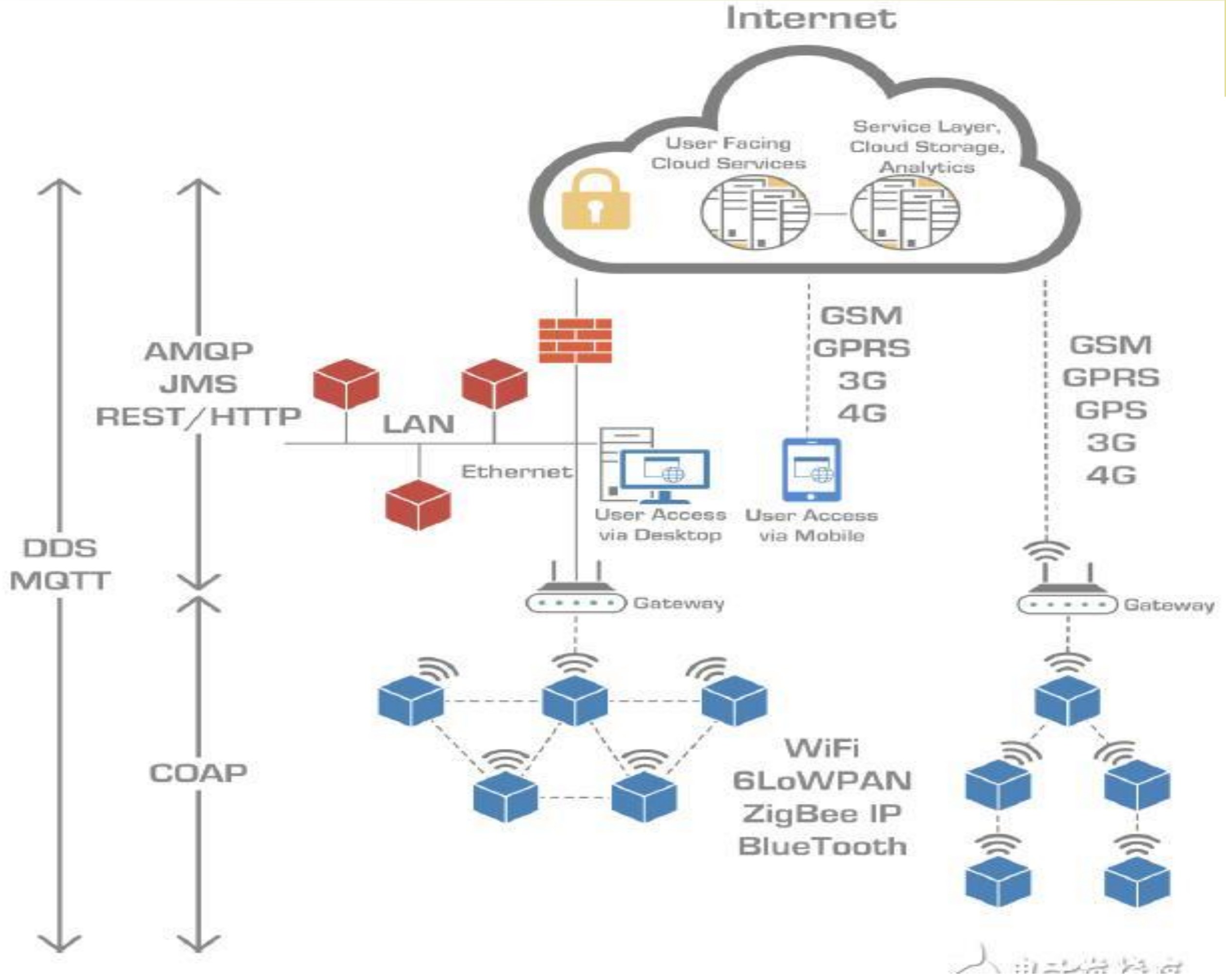
Outline

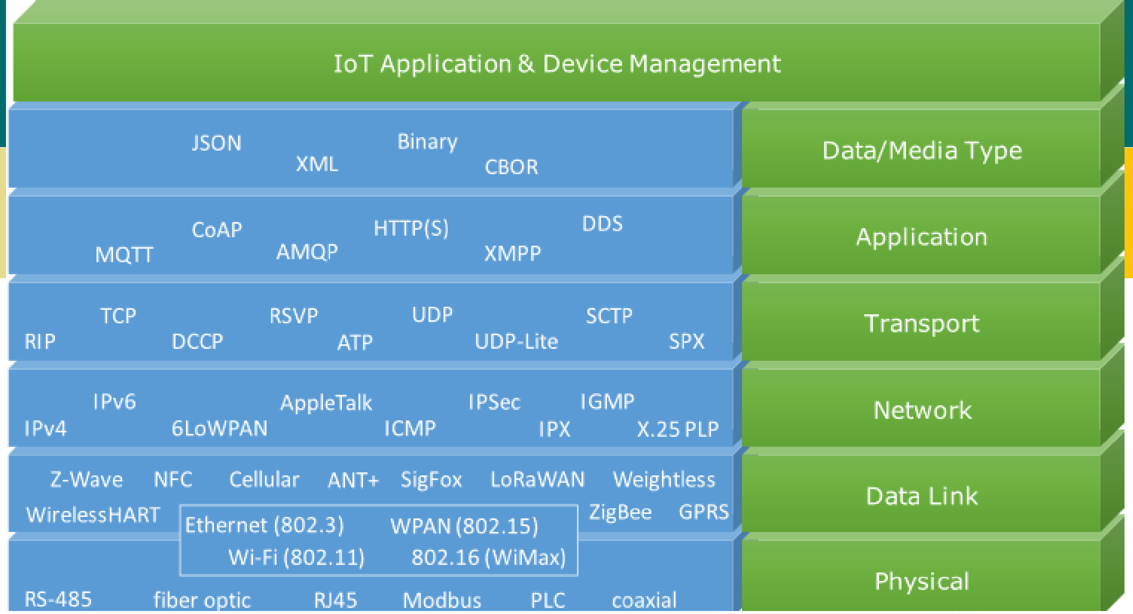
- Review: Wireless Communications in IoT
- Overview of IoT Application Protocols
- Survey of IoT Platform Protocols
- Open Source IoT EcoSystem
- List of IoT Application Protocols
- Web of Things (WoT)
- HTTP/REST API
- CoAP
- MQTT
- Comparison

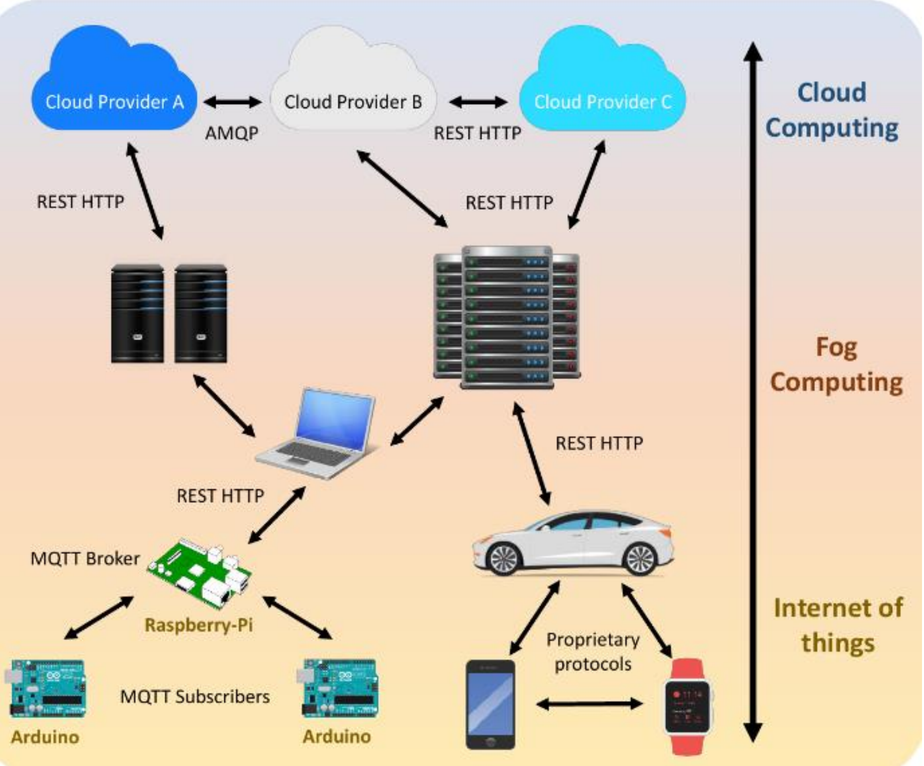
Wireless Communications in IoT

Technology	Standard	Frequency	Penetration	Range	Max Data Rate	Channel Bandwidth	Chipset Cost
NFC/RFID	ISO/ICE 18092	13.56 MHz	High	<20 cm	424 kbps	106–424 Mbps	$0.1+
Bluetooth	IEEE 802.15	2.4/2.5 GHz	Low	50–100 m	2 Mbps	2 MHz	$5+
Wi-Fi	IEEE 802.11	2.4/5.0 GHz	Low	100 m	54 Mbps	22 MHz	$1.5-30+
Zigbee	IEEE 802.15.4	868/915 MHz, 2.4 GHz	Low/High	<1 km	250 kbps	2 MHz	$2-20+
DASH7	ISO/IEC 18000-7	433/868/915 MHz	High	0–5 km	167 kbps	up to 1.75 MHz	$3.00+
Weightless	Weightless P/N/W	Multiple	Low/High	5 km	100 kbps	200 Hz–12.5 KHz	∼$2.00
LoRa	Various	868/915 MHz	Low	25 km	50 kbps	125/250/500 kHz	∼$2.00
Ingenu-RPMA	Ingenu-RPMA	2.4 GHz	Low	15 km	20 kbps	1 MHz	rental
SigFox	SigFox	915–928 MHz	Low/High	40 km	100 bps	100 Hz	$0.25+
3G	UMTS/W-CDMA	0.4–3 GHz	Low/High	5–35 km	0.38–21.6 Mbps	3.6–21 Mbps	varies
4G/LTE	3GPP-LTE	0.6–6 GHz	Low/High	5–100 km	100–300 Mbps	100 Mbps+	$6.5+
5G	5GTF/5G-SIG	0.6–4/100 GHz	Low/High	5–150 km	10 Gbps	500 Mbps+	$70+

Overview of IoT Application Protocols







Survey of IoT Platform Protocols [1]

IoT Platform, *Year of First General Availability (GA)*	Protocol
Azure IoT Hub [166], *2014*	HTTP(S), MQTT, MQTT over WebSocket, AMQP, AMQP over WebSocket *custom protocols transmit via a gateway*
Google IoT Core [167], *2018*	HTTP(S), MQTT *custom protocols transmit via a gateway*
IBM Watson IoT [168] *2014*	HTTP(S), MQTT
AWS IoT Core [169] *2015*	HTTP(S), MQTT, MQTT over WebSocket, WebSocket
Alibaba IoT [170] 2015	HTTP(S), CoAP, MQTT, MQTT over WebSocket, WebSocket, *support network types: 3G, 4G, NB-IoT & LoRa*
Oracle IoT [171] *2016*	HTTP(S), CoAP, MQTT, AMQP, XMPP, WebSocket
Siemens MindSphere [172] *2016*	HTTP(S), CoAP, MQTT, AMQP, XMPP, *supports wide range of device protocols via field gateways (e.g. MindConnect) such as OPC UA, LoRaWAN, Modbus, 6LoWPAN, LwM2M*
Bosch IoT Hub [173] *2017*	HTTP(S), MQTT, AMQP, LoRaWAN
Cisco Kinetic [174] *2017*	HTTP(S), MQTT, AMQP, WebSocket *custom protocols transmit via a gateway (e.g. Cisco IoT Gateway)*
Eclipse Hono [175] *2018*	HTTP(S), CoAP, MQTT, AMQP *uses AMQP 1.0 as primary messaging protocol custom protocols transmit via a gateway*

Survey of IoT Platform Protocols [2]

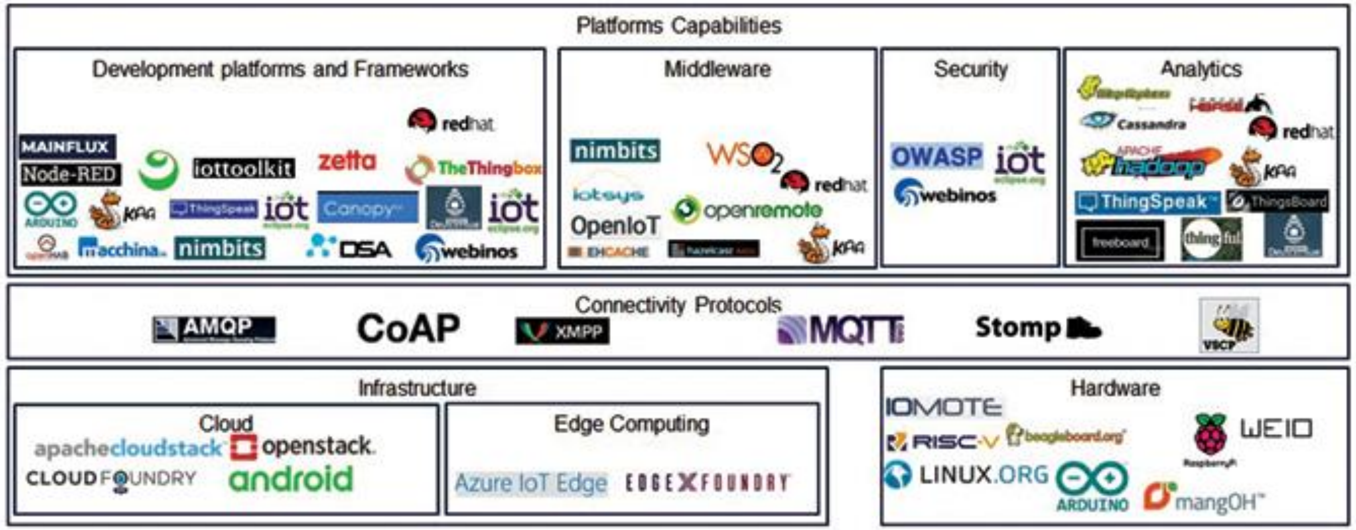
Features	AWS	Microsoft Azure	Google Cloud IoT	IBM Watson IoT	Oracle IoT
Security	Link Encryption (TLS), Authentication (Sig V4, X.509)	Link Encryption (SSL/TSL)	SSL/TLS	Link Encryption (TLS), Authentication (IBM cloud SSO), Identity Management (LDAP)	REST API
Data analytics	Real Time analytics (Rule engine, Kinesis, AWS Lambda)	Real Time analytics	Real Time analytics (Cloud IoT Core)	Real Time analytics (IBM IoT Real time insights)	Real Time analytics
Protocols	MQTT, HTTP1.1	MQTT, HTTP, AMQP	MQTT	MQTT, HTTPS	MQTT, HTTP
Visualization tool	AWS IoT dashboard	web portal	Google data studio (Dashboard)	web portal	web portal
Data format	JSON	JSON	JSON	JSON, CSV	CSV, REST API
Application Environment	Java, C, NodeJs, Javascript, Python, SDK for Arduino, iOS, Android	.Net, UWP, Jave, C, NodeJS, Ruby, Android,iOS	Go,Java, Python, .NET, NodeJS, php, Ruby	C#, C, Python, Java, NodeJS	Java, iOS, Javascript, C, Android

Survey of IoT Platform Protocols [3]

IoT Software	Integration	Data collection	Analyses	Visualization	Data Base
ThingsBoard	REST APIs, MQTT APIs	HTTP, MQTT, OPC-UA,CoAP	Real time analytics (Apache Spark, kafka),	Yes	PostgreSQL, Cassandra, HSQLDB
Kaa	Portable SDK available to integrate any particular platform, REST API	MQTT, CoAP, XMPP, TCP, HTTP, WiFi, Ethernet, Zigbee	Real time	Yes (Doesn't have own dashboards) NoSQL	MongoDB, Cassandra, Hadoop, Oracle
WSo2	REST APIs	HTTP, WSO2 ESB, MQTT	Yes, WSO2 Data Analytics Server	Yes	Oracle, PostgreSQL, MySQL, or MS SQL
Site Where	REST API	MQTT, AMQP, Stomp, WebSockets, and direct socket connections	Real-time analytics (Apache Spark)	No	MongoDB, HBase, InfluxDB
Thing Speak	REST API, MQTT APIs	HTTP	MATLAB Analytics	No	MySQL
DeviceHive	REST API, MQTT APIs	REST API, WebSockets or MQTT	Real-time analytics (Apache Spark)	Yes (Doesn't have own dashboards)	PostgreSQL, SAP Hana DB
Zetta	REST APIs	HTTP	Using Splunk	No	Unknown
Distributed Services Architecture (DSA)	REST APIs	HTTP	No	No	ETSDB Embedded Time Series
Thinger.io	REST APIs	MQTT, CoAP and HTTP	Yes	No	MongoDB

Open Source IoT EcoSystem

- There is a need for organizations to provide a validated, modular, flexible IoT solution built to be open, interoperable and cost-effective.
- The solution should deliver end-to-end open source IoT that addresses enterprise level needs.
- The characteristics of open source IoT architecture are:
  - Loosely coupled, modular and secure
  - Platform independent
  - Scalable, flexible and can be deployed anywhere
  - Based on open standards
  - Streaming analytics and machine learning
  - Open and interoperable on the hybrid cloud
  - Application agility and integration
  - No vendor lock-in, no rigid architectures or proprietary formats and components



Ref:
https://www.opensourceforu.com/2020/02/leveraging-open-source-tools-for-iot/
https://vizah.ch/en/developing-iot-applications-best-technologies-and-tools-for-iot-developers/

List of IoT Application Protocols (1/2)

- **CoAP (Constrained Application Protocol):** RFC 7252
  - CoAP makes use of the UDP protocol RESTful architecture (based on HTTP protocol).
  - It is used within mobiles and social network based applications.
- **MQTT:** OASIS
  - This messaging protocol is used for remote monitoring in IoT.
  - MQTT connects devices and networks with applications and middleware using hub-and-spoke architecture.
  - The MQTT protocol provides efficient information routing functions to small, cheap, low-memory and power consuming devices in vulnerable and low bandwidth based networks.
- **XMPP (Extensible Messaging and Presence Protocol):** RFC3920/RFC6120 - (https://xmpp.org/)
  - P2P or Pub/Sub methodology.
  - This is a communications IoT protocol for message-oriented middleware based on the **XML** language.
  - It enables real-time exchange of structured yet extensible data between any two or more network entities.
  - XMPP enables messaging applications to attain authentication, access control, hop-by-hop and end-to-end encryption.
- **AMQP (Advanced Message Queuing Protocol):** ISO/IEC 19464:2014 - (https://www.amqp.org/)
  - It supports reliable communication via message delivery assurance primitives like 'at-most once', 'at least once' and 'exactly once'.
  - This protocol enables client applications to talk to the broker, route and store messages within a broker, and interact with the AMQP model.
- **DDS (Data Distribution Service):** OMG
  - This is used for real-time machine-to-machine communication.
  - Middleware connectivity framework
  - It enables scalable, real-time, dependable, high-performance and interoperable data exchange via the publish-subscribe methodology.
  - DDS can be deployed in platforms ranging from low-footprint devices to the cloud, and supports efficient bandwidth usage as well as the agile orchestration of system components.
- Others: OPC UA (Open Platform Communications—Unified Architecture), NDN (Named Data Networking) for IoT

List of IoT Application Protocols (2/2)

- **VSCP (Very Simple Control Protocol):** https://www.vscp.org/
  - Device discovery and identification.
  - Device configuration.
  - Autonomous device functionality.
  - Secure update of device firmware
  - A solution from sensor to UI
  - It uses CAN, RS-232, Ethernet, TCP/IP, MQTT, 6LowPan or whatever as it's transport mechanism and work over cable and over the air.

Stomp

The Simple Text Oriented Messaging Protocol

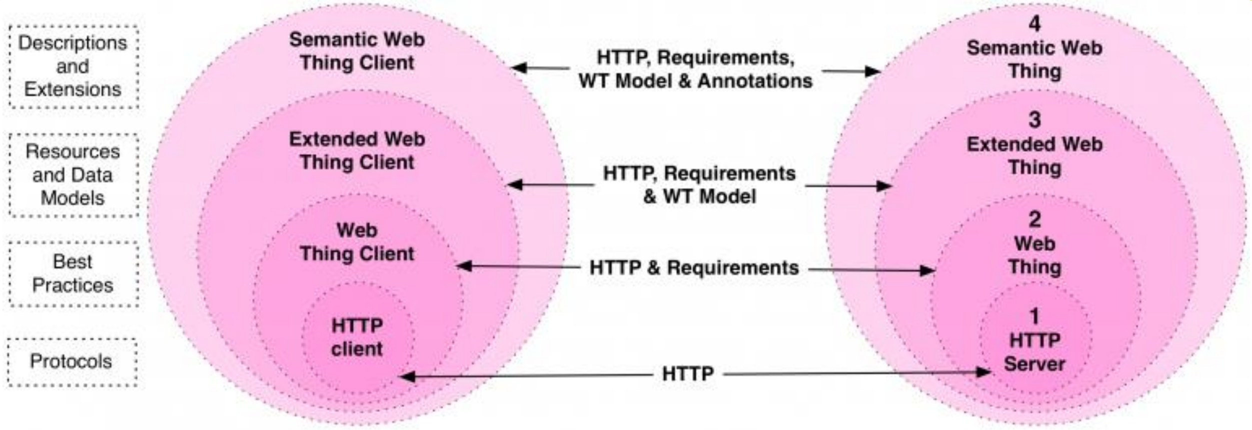
- **STOMP (Simple Text Oriented Messaging Protocol):** https://stomp.github.io/
  - STOMP provides an interoperable wire format
  - STOMP clients can communicate with any STOMP message broker to provide easy and widespread messaging interoperability among many languages, platforms and brokers.
  - STOMP is a very simple and easy to implement protocol, coming from the HTTP school of design; the server side may be hard to implement well, but it is very easy to write a client to get yourself connected. For example you can use Telnet to login to any STOMP broker and interact with it!
  - Many developers have told us that they have managed to write a STOMP client in a couple of hours to in their particular language, runtime or platform into the STOMP network. So if your favored language/runtime of choice does not offer a good enough STOMP client don't be afraid to write one.
- **QUIC (Quick UDP Internet Connections):** https://quicwg.org/
  - A flexible multiplexed and secure general-purpose transport protocol that supports multiplexed streams.

QUIC

Ref:
http://dave.thehorners.com/aboutme/cooking-recipes/499-messaging-messagequeue-pubsub-stomp-amqp-mqtt

Web of Things (WoT): Model (1/2)

Ref: https://www.w3.org/blog/2017/01/web-thing-model-member-submission/

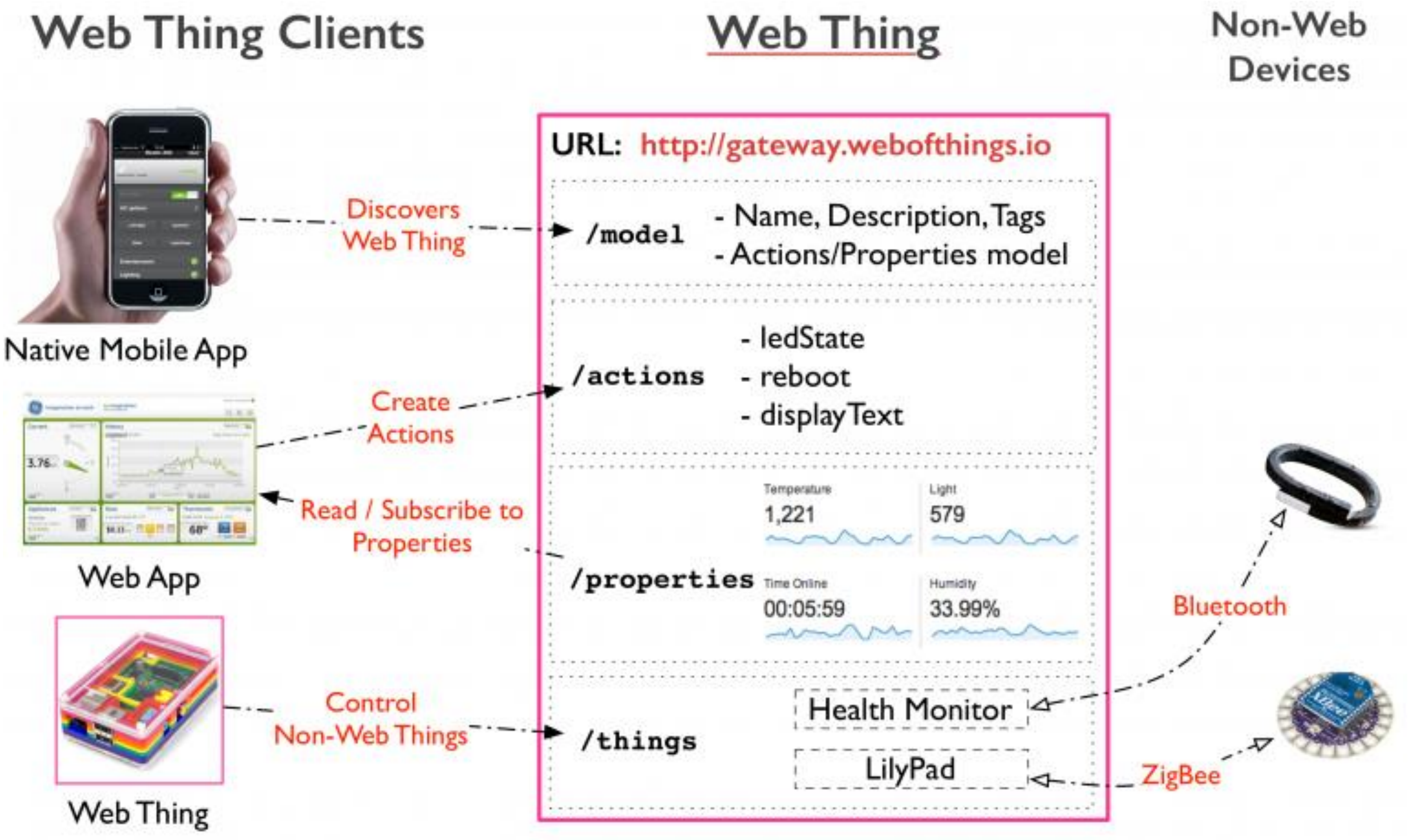


Source: Building the Web of Things: book.webofthings.io
Creative Commons Attribution 4.0

- **Things** – A web Thing can be a gateway to other devices that don't have an internet connection. This resource contains all the web Things that are proxied by this web Thing. This is mainly used by clouds or gateways because they can proxy other devices.
- **Model** – A web Thing always has a set of metadata that defines various aspects about it such as its name, description, or configurations.
- **Properties** – A property is a variable of a web Thing. Properties represent the internal state of a web Thing. Clients can subscribe to properties to receive a notification message when specific conditions are met; for example, the value of one or more properties changed.
- **Actions** – An action is a function offered by a web Thing. Clients can invoke a function on a web Thing by sending an action to the web Thing.
  - ✓ Examples of actions are "open" or "close" for a garage door, "enable" or "disable" for a smoke alarm, and "scan" or "check in" for a bottle of soda or a place.
  - ✓ The direction of an action is usually from the client to the web Thing.
  - ✓ Actions represent the public interface of a web Thing and properties are the private parts. Much like in any programming languages, you can access the public interface, and whatever is private remains accessible only for privileged parties, like the instance itself or, in this case, the web Thing. But limiting access to actions – that is, the public interface – also allows you to implement various control mechanisms for external requests such as access control, data validation, updating a several properties atomically, and the like.

Web of Things (WoT): Model (2/2)

Ref: https://www.w3.org/blog/2017/01/web-thing-model-member-submission/

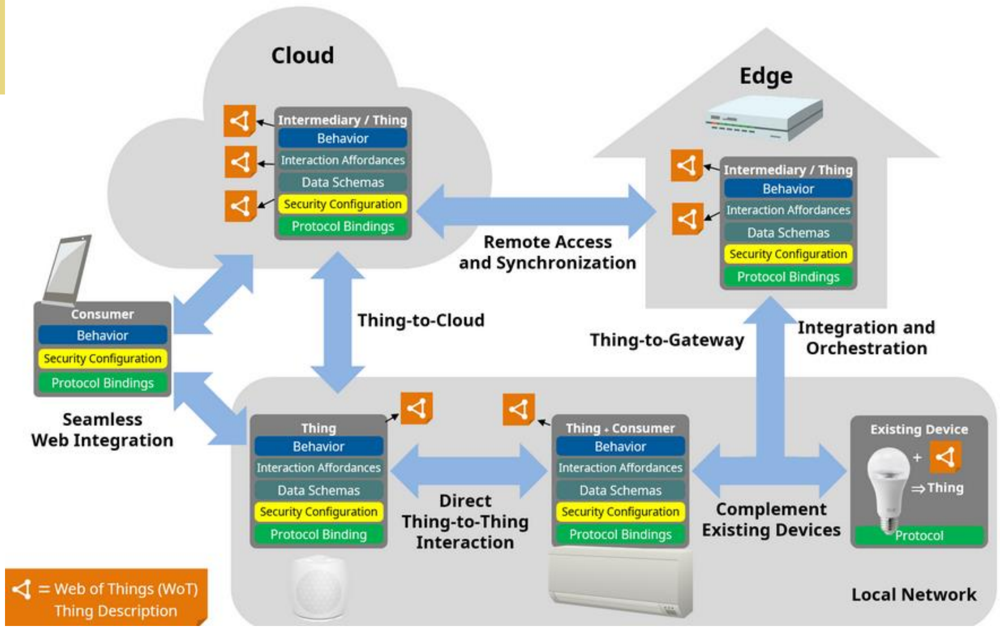


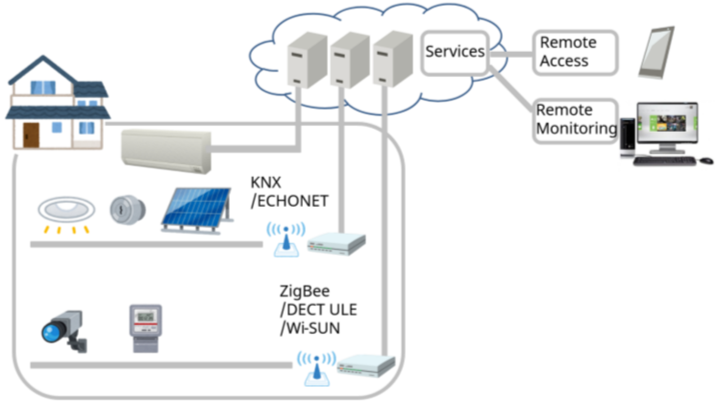
Source: Building the Web of Things: book.webofthings.io
Creative Commons Attribution 4.0

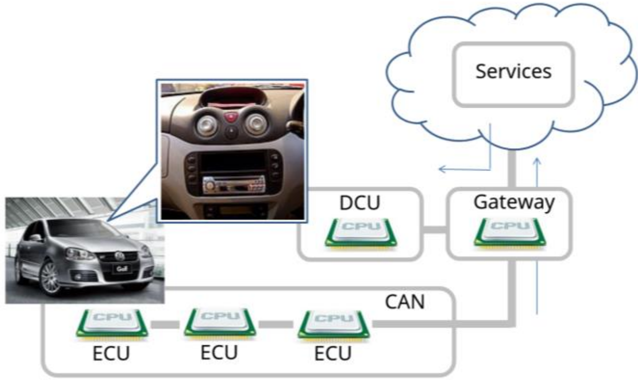
Web of Things (WoT): Architecture

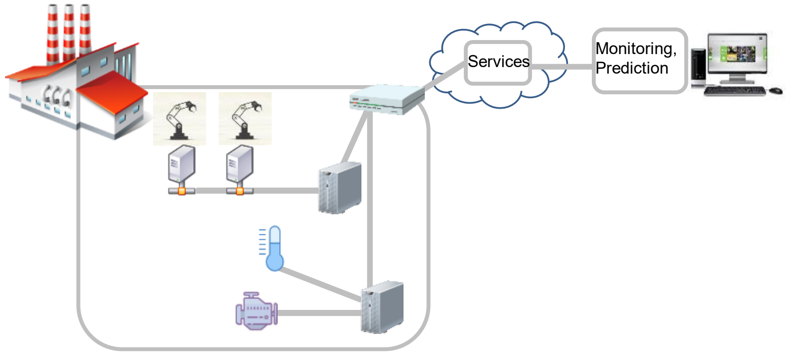
Ref: https://www.w3.org/2020/04/pressrelease-wot-rec.html.en

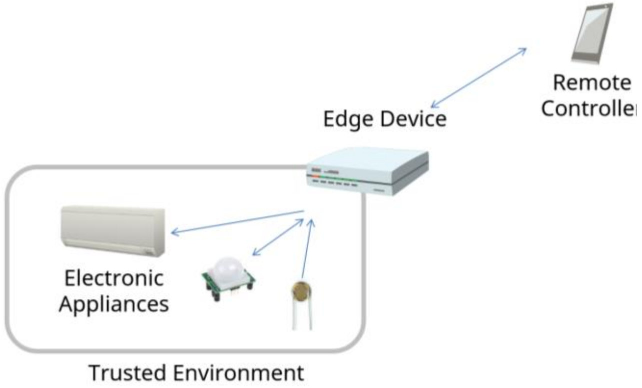
- The Web of Things is applicable to multiple IoT domains, including Smart Home, Industrial, Smart City, Retail, and Health applications, where usage of the W3C WoT standards can simplify the development of IoT systems that combine devices from multiple vendors and ecosystems.

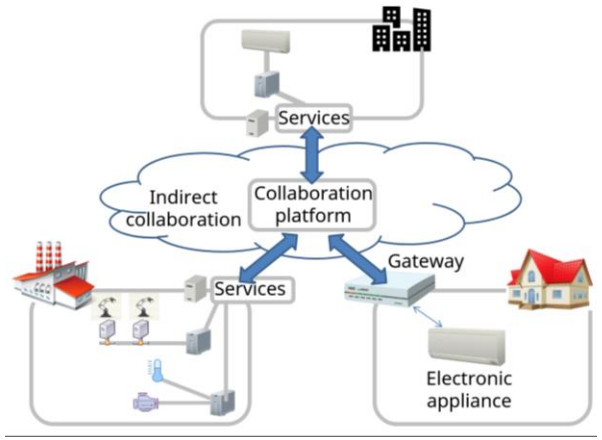












Web Thing API (1/2)

Ref: https://iot.mozilla.org/wot/

- The Web Thing Description provides a vocabulary for describing physical devices connected to the World Wide Web in a machine readable format with a default JSON encoding.

- Common device capabilities can be specified using optional semantic annotations.

- The Web Thing REST API and Web Thing WebSocket API allow a web client to access the properties of devices, request the execution of actions and subscribe to events representing a change in state.

A simple Thing Description


```
{
  "id": "https://mywebthingserver.com/things/switch",
  "title": "On/Off Switch",
  "description": "A web connected switch",
  "properties": {
    "on": {
      "title": "On/Off",
      "type": "boolean",
      "description": "Whether the lamp is turned on",
      "links": [{"href":
"/things/switch/properties/on"}]
    }
  }
}
```

Web Thing API (2/2)

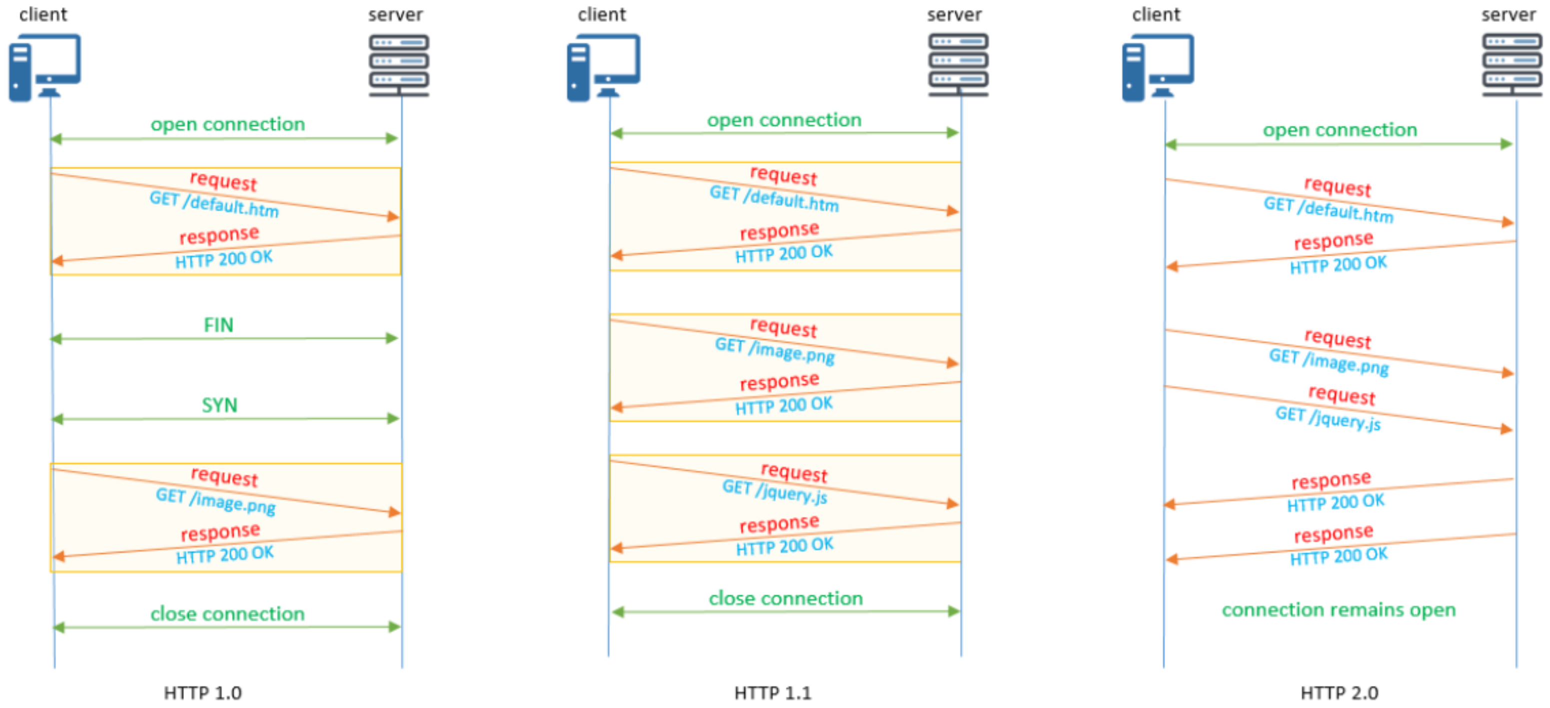
```
{
  "@context": "https://iot.mozilla.org/schemas/",
  "@type": ["Light", "OnOffSwitch"],
  "id": "https://mywebthingserver.com/things/lamp",
  "title": "My Lamp",
  "description": "A web connected lamp",
  "properties": {
    "on": {
      "@type": "OnOffProperty",
      "type": "boolean",
      "title": "On/Off",
      "description": "Whether the lamp is turned on",
      "links": [{"href": "/things/lamp/properties/on"}]
    },
    "brightness" : {
      "@type": "BrightnessProperty",
      "type": "integer",
      "title": "Brightness",
      "description": "The level of light from 0-100",
      "minimum" : 0,
      "maximum" : 100,
      "links": [{"href": "/things/lamp/properties/brightness"}]
    }
  },
 "actions": {
    "fade": {
      "@type": "FadeAction",
      "title": "Fade",
      "description": "Fade the lamp to a given level",
      "input": {
        "type": "object",
        "properties": {
          "level": {
            "type": "integer",
            "minimum": 0,
            "maximum": 100
          },
          "duration": {
            "type": "integer",
            "minimum": 0,
            "unit": "milliseconds"
          }
        }
      },
      "links": [{"href": "/things/lamp/actions/fade"}]
    }
  },
```

```
  "events": {
    "overheated": {
      "title": "Overheated",
      "@type": "OverheatedEvent",
      "type": "number",
      "unit": "degree celsius",
      "description": "The lamp has exceeded its safe operating temperature",
      "links": [{"href": "/things/lamp/events/overheated"}]
    }
  },
  "links": [
    {
      "rel": "properties",
      "href": "/things/lamp/properties"
    },
    {
      "rel": "actions",
      "href": "/things/lamp/actions"
    },
    {
      "rel": "events",
      "href": "/things/lamp/events"
    },
    {
      "rel": "alternate",
      "href": "wss://mywebthingserver.com/things/lamp"
    },
    {
      "rel": "alternate",
      "mediaType": "text/html",
      "href": "/things/lamp"
    }
  ]
}
```

Ref: https://iot.mozilla.org/wot/

HTTP (HyperText Transfer Protocol)

Client (Request)/ Server (Response)

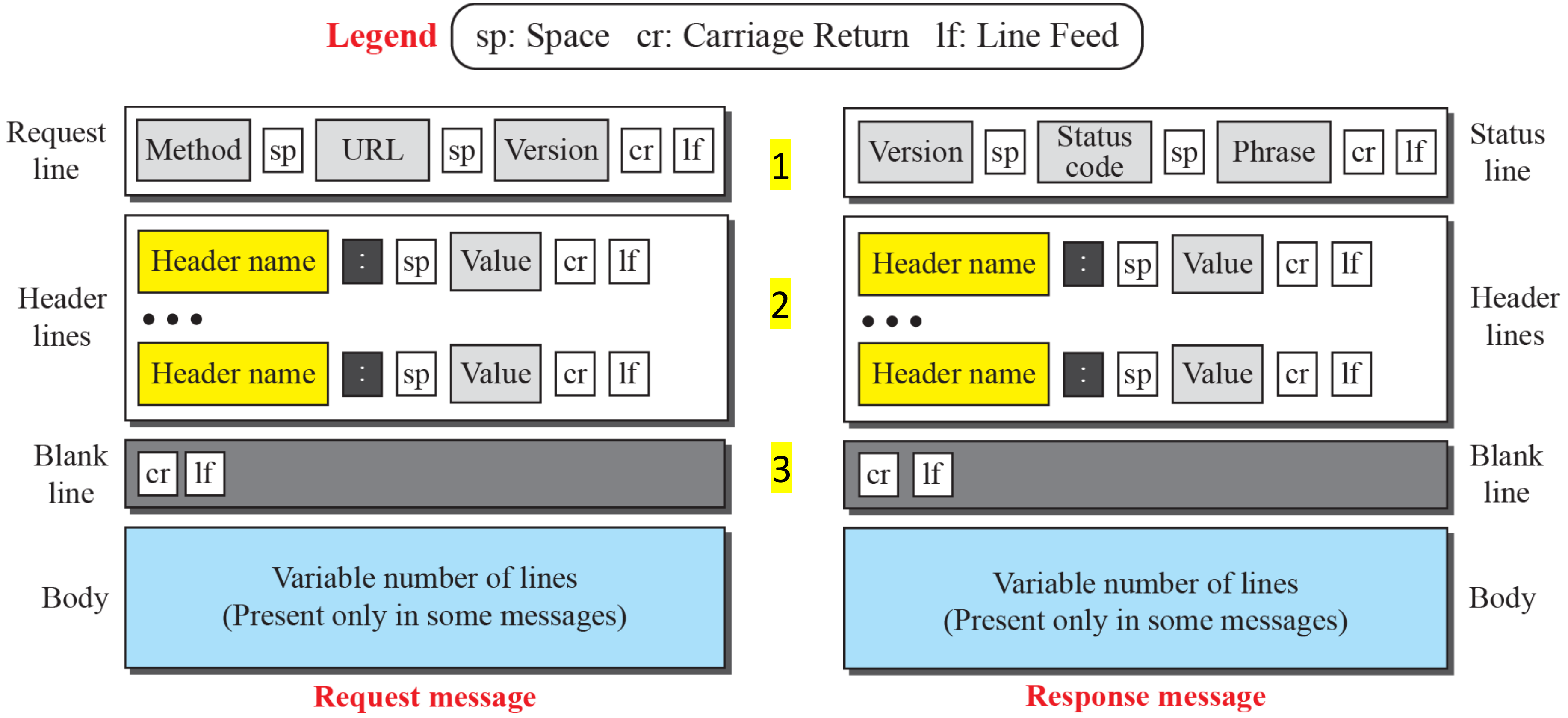


HTTP Client: Web Browser
HTTP Server: Web Server

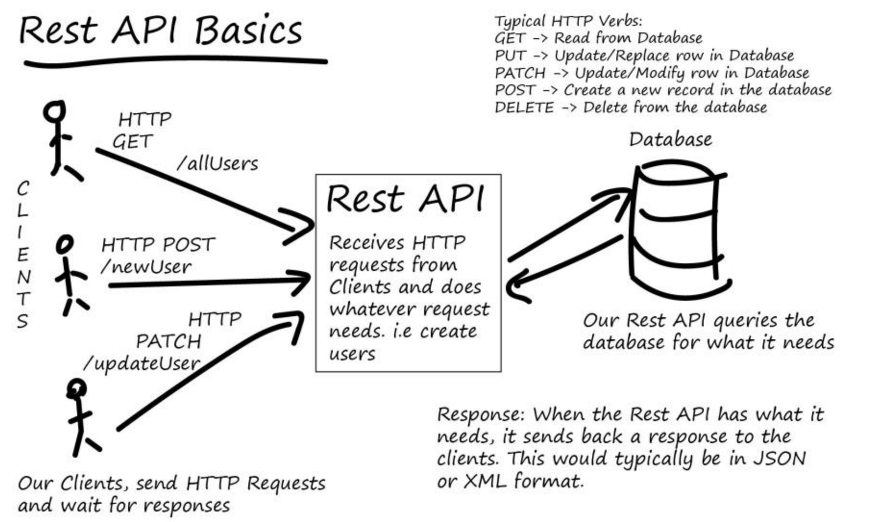
Web Server:
https://httpd.apache.org

- An application-level and stateless protocol based on request/response and client/server model for data communication over the World Wide Web.
- TCP Connection, port 80
- One of the key features of HTTP is content negotiation of data representation.
- This enables different heterogeneous devices built independently of the data to be shared.

The Format of Two Message Types



- REST was introduced and defined in 2000 by Roy Fielding in his doctoral dissertation.
- It is NOT standard, but REST is an architectural style.
  Note: Standards like HTTP, URL
  - Resource Representations: XML/HTML/GIF/JPEG/etc.
  - Resource Types, MIME Types: text/xml, text/html, image/gif, image/jpeg, etc.
- The motivation for REST was to capture the characteristics of the Web which made the Web successful
  - URI Addressable resources
  - HTTP Protocol
  - Make a Request – Receive Response – Display Response
  - Stateless: simple request/response. No conversational state. Easy to scale. The request must contain all that is required for the reply to be computed.
- HATEOAS (Hypermedia As The Engine Of Application State)



- Exploits the use of the HTTP protocol
  - HTTP POST
  - HTTP GET
  - HTTP PUT
  - HTTP DELETE

Ref: https://digital-goal.net/foundation-of-restful-api-architecture/

REST (Representational State Transfer)

Examples (JSON-LD)

• `GET /basement/water/temperature`	`200 OK`
	`application/text`
	`40.5 F`

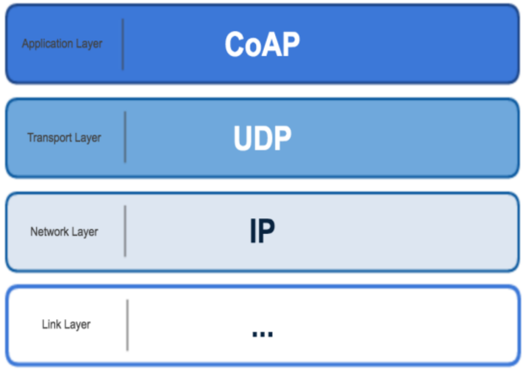
• `GET /basement/water/volume`	`200 OK`
	`application/text`
	`200 G`

JSON-LD format specification for linked open data to interoperate at web-scale

Ref:
https://www.w3.org/TR/json-ld-api/
https://www.w3.org/2018/json-ld-wg/

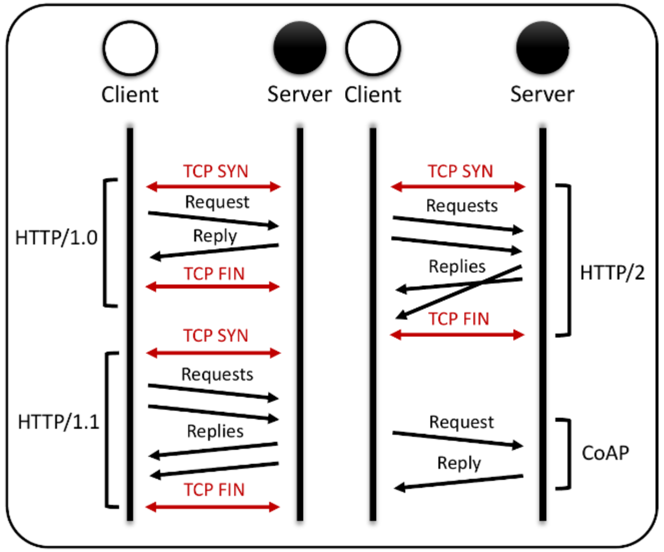
CoAP (Constrained Application Protocol)

- **REST-based web transfer protocol**
- Manipulates Web resources using the same methods as HTTP: GET, PUT, POST, and DELETE
- Subset of HTTP functionality re-designed for low power embedded devices such as sensors (for IoT and M2M)
- TCP overhead is too high and its flow control is not appropriate for short-lived transactions
- UDP has lower overhead and supports multicast



Application (CoAP, XML)
Security (DTLS)
Transport (UDP)
Network (IPv6)
PHY/MAC (IEEE 802.15.4)

DTLS (Datagram Transport Layer Security)



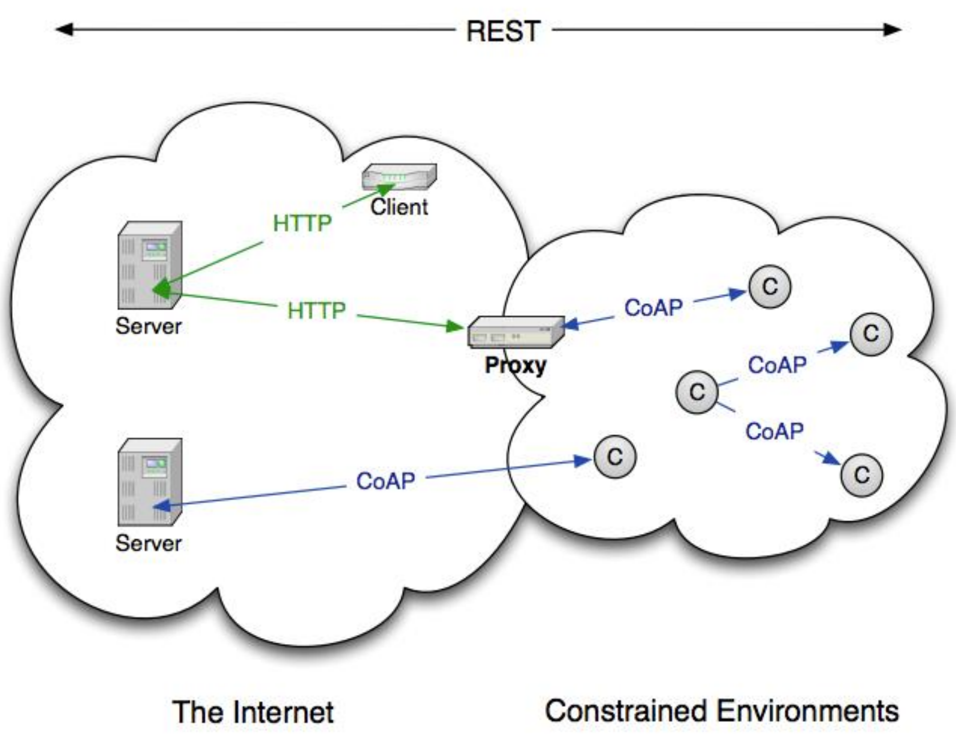
CoAP (Constrained Application Protocol)

- Four message types:
  - **Confirmable** – requires an ACK
  - **Non-confirmable** – no ACK needed
  - **Acknowledgement** – ACKs a Confirmable
  - **Reset** - indicates a Confirmable message has been received but context is missing for processing
- CoAP provides reliability without using TCP as transport protocol
- CoAP enables asynchronous communication
  - e.g., when CoAP server receives a request which it cannot handle immediately, it first ACKs the reception of the message and sends back the response in an off-line fashion
- Also supports multicast and congestion control

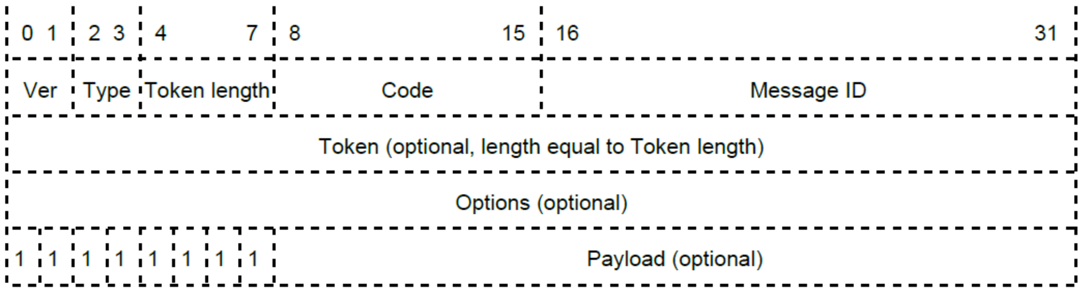
Copper plugin in Firefox

Californium

CoAP (Constrained Application Protocol)



- A RESTful protocol
- Both synchronous and asynchronous
- For constrained devices and networks
- Specialized for M2M applications
- Easy to proxy to/from HTTP
- Implement: https://libcoap.net
- Benchmark: https://www.eclipse.org/californium

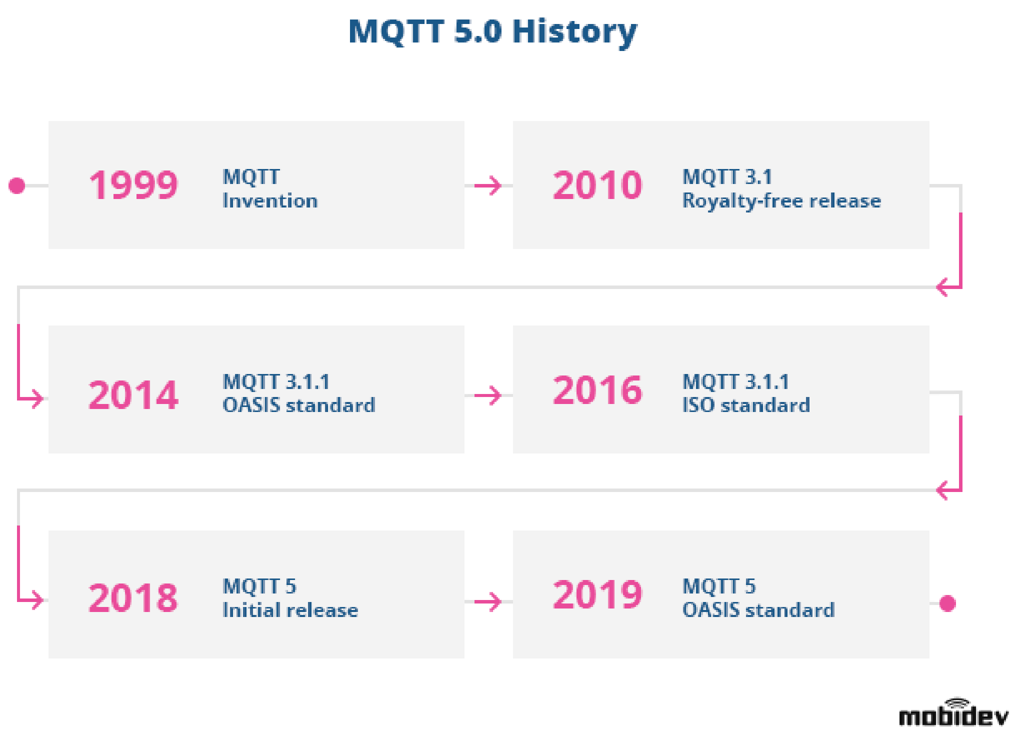


Learn more:
https://tools.ietf.org/id/draft-keranen-t2trg-rest-iot-05.html

MQTT (Message Queuing Telemetry Transport)

Introduction

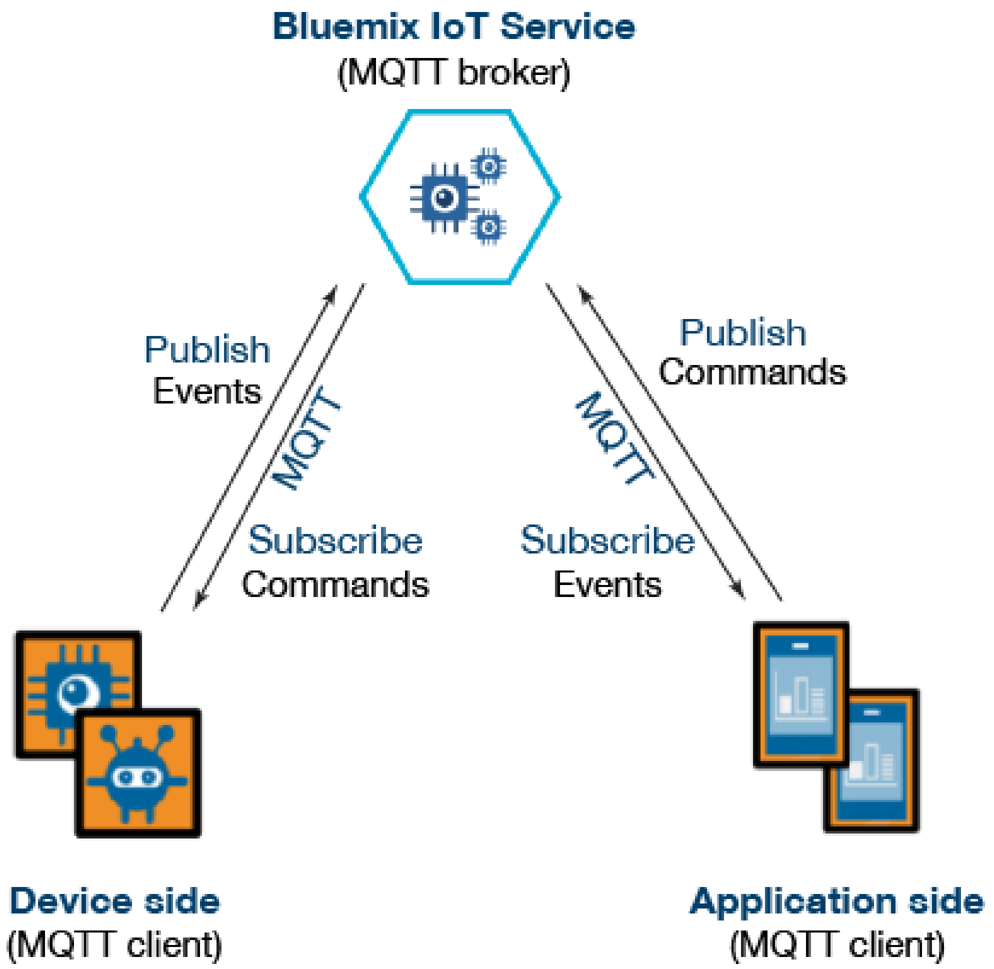
- MQTT was invented by Andy Stanford-Clark (IBM) and Arlen Nipper (Arcom, now Cirrus Link) back in 1999, where their use case was to create a protocol for minimal battery loss and minimal bandwidth connecting oil pipelines over satellite connections.
- They specified the following goals, which the future protocol should have:
  - Simple to implement
  - Unreliable networks
  - Provide a Quality of Service Data Delivery
  - Lightweight and Bandwidth Efficient
  - Data Agnostic
  - Continuous Session Awareness



MQTT (Message Queuing Telemetry Transport)

Architecture

- In a nutshell, MQTT consist of three parts:
  - Broker
  - Subscribers
  - Publishers
- You can send anything as a message; up to 256 MB

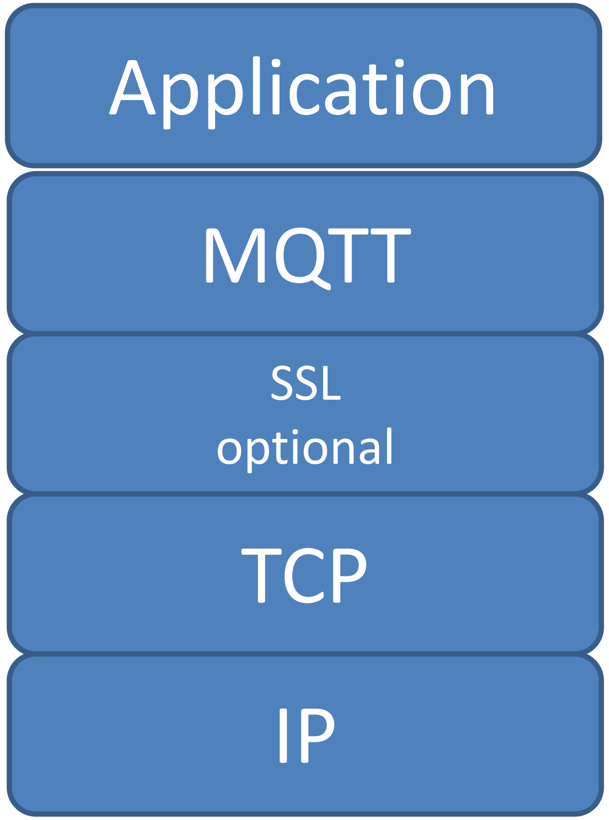


https://www.reddit.com/r/homeautomation/comments/a1gest/diy_home_automation_esp32_raspberry_pi_node_red/

MQTT (Message Queuing Telemetry Transport)

Protocol Stack

TCP/IP Port: 1883
When running over SSL, TCP/IP port 8883
SSL: Secure Socket Layer (encryption)

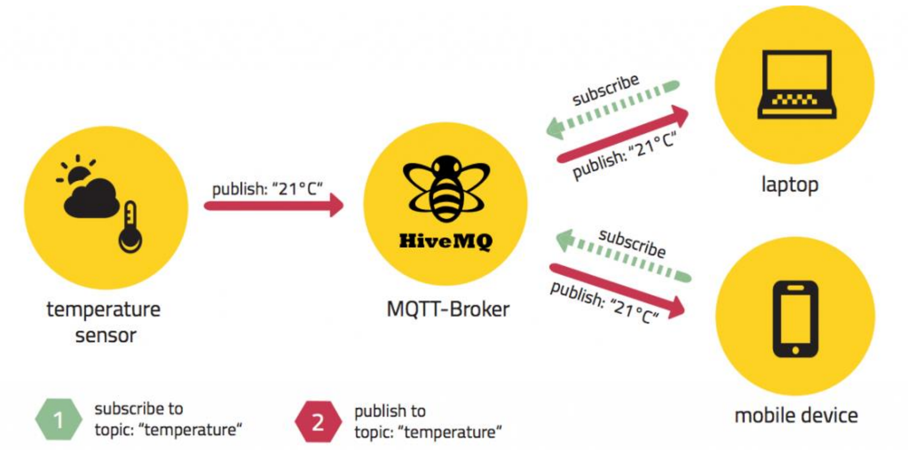


Learn more about MQTT
https://www.youtube.com/c/HiveMQ

MQTT (Message Queuing Telemetry Transport)

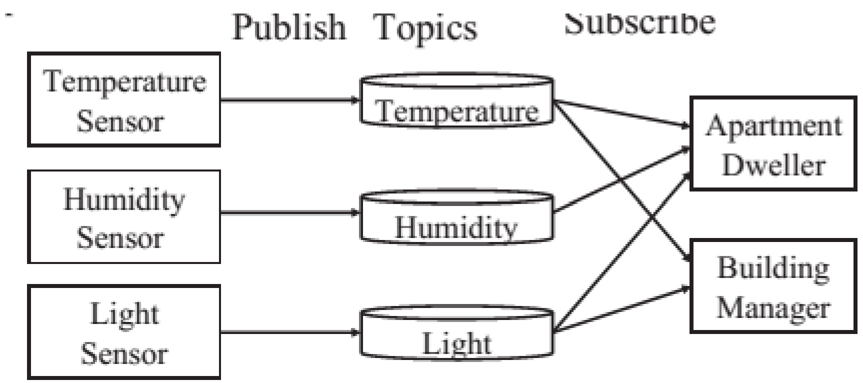
Publish/Subscribe Concept

- **Decoupled in space and time:**
  - The clients do not need each others IP address and port (space) and they do not need to be running at the same time (time).
- **The broker's IP and port must be known by clients**
  - Namespace hierarchy used for topic filtering
  - It may be the case that a published message is never consumed by any subscriber
- **Broker**
  - Receives all the messages
  - Filters the messages
  - Publishes the messages to all subscribed clients



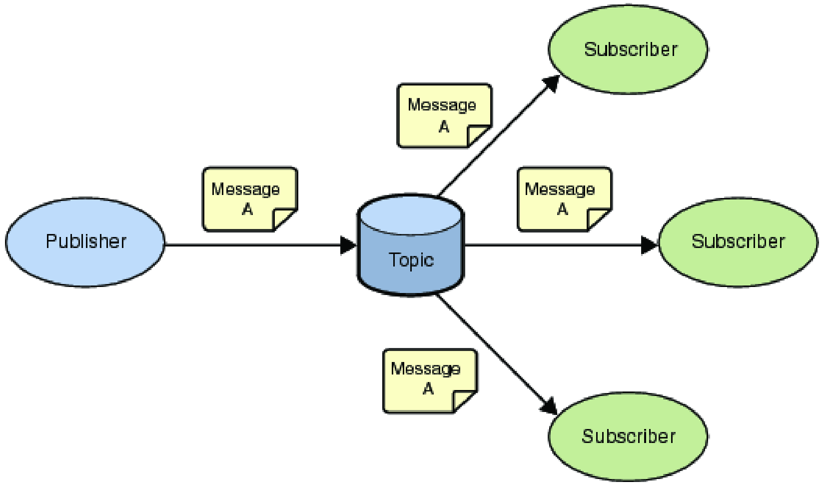
Publish/Subscribe Concept

- **Topics/Subscriptions:**
  - Messages are published to topics.
  - Client can subscribe to a topic or a set of related topics
- **Publish/Subscribe**:
  - Clients can subscribe to topics or publish to topics



[Ref: ces570-15@www.cse.wustl.edu/]

Publish / Subscribe Messaging (One to Many)



Producer: Publishes a message (publication) on a topic (subject)
Consumer: Subscribes (makes a subscription) for messages on a topic (subject)

A message server matches publications to subscriptions:

- If none of them match the message is discarded
- If one or more matches the message is delivered to each matching consumer

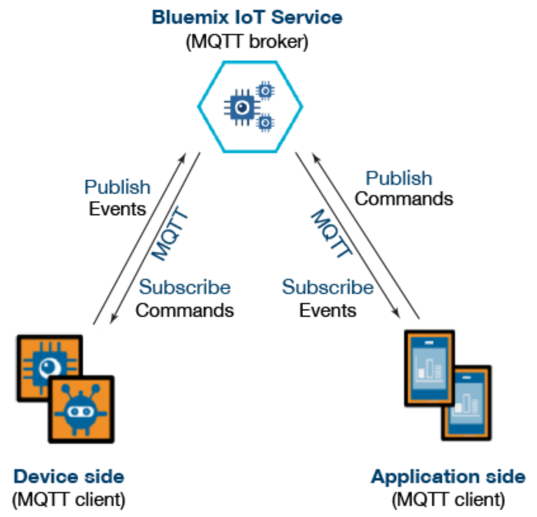
Publish / Subscribe has three important characteristics:

1. It decouples message senders and receivers, allowing for more flexible applications
2. It can take a single message and distribute it to many consumers
3. This collection of consumers can change over time, and vary based on the nature of the message.

MQTT (Message Queuing Telemetry Transport)

Example

- Clients connect to a "Broker"
- Clients subscribe to topics e.g.,
  - `client.subscribe('toggleLight/1')`
  - `client.subscribe('toggleLight/2')`
  - `client.subscribe('toggleLight/3')`
- Clients can publish messages to topics:
  - `client.publish('toggleLight/1','toggle');`
  - `client.publish('toggleLight/2','toggle');`



- All clients receive all messages published to topics they subscribe to
- **Messages can be anything**
  - Text
  - Images
  - etc.

MQTT (Message Queuing Telemetry Transport)

Topics: General Rules

- Each published data specifies a topic
- Each subscriber subscribed to that topic will receive it
- Topics are case-sensitive:

`home/office/lamp <> home/office/LAmp`

- Topic format:

```
separator

/home/rooms/kitchen/temperature

sub-topic  sub-topic  sub-topic  sub-topic
```

https://www.youtube.com/watch?v=jZL4uohxuPc&t=7s

Topics: Wildcards

- When a client subscribes to a topic, it can subscribe to the exact topic of a published message or it can use wildcards to subscribe to multiple topics simultaneously.
- A wildcard can only be used to subscribe to topics, not to publish a message.
- There are two different kinds of wildcards: *single-level* and *multi-level*.

(1) **Single Level**: +

- ✓ To replace one topic level
- ✓ Any topic matches a topic with single-level wildcard if it contains an arbitrary string instead of the wildcard.

(2) **Multi Level**: #

- ✓ To cover many topic levels
- ✓ The hash symbol represents the multi-level wild card in the topic and must be placed as the last character in the topic and preceded by a forward slash.

Topics: Beginning with $

- Generally, you can name your MQTT topics as you wish.
- However, there is one exception: **Topics that start with a $ symbol have a different purpose.**
- These topics **are not part** of the subscription when you subscribe to the multi-level wildcard as a topic (#).
- **The $-symbol topics are reserved for internal statistics of the MQTT broker.**
- Clients cannot publish messages to these topics.
- At the moment, there is no official standardization for such topics.
- Commonly, **$SYS/** is used for all the following information, but broker implementations varies.
- One suggestion for $SYS-topics is in the MQTT GitHub wiki [1]. Here are some examples:
  - ```
    $SYS/broker/clients/connected
    $SYS/broker/clients/disconnected
    $SYS/broker/clients/total
    $SYS/broker/messages/sent
    $SYS/broker/uptime
    ```

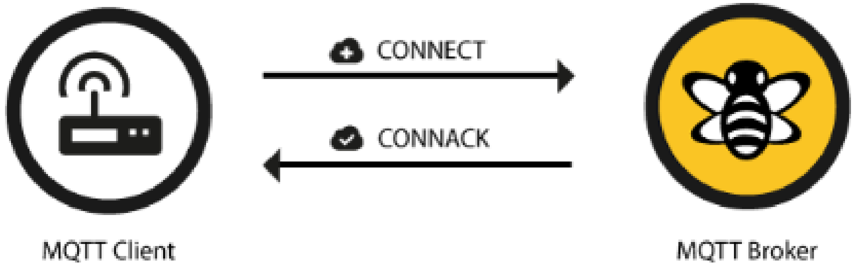
Mosquitto

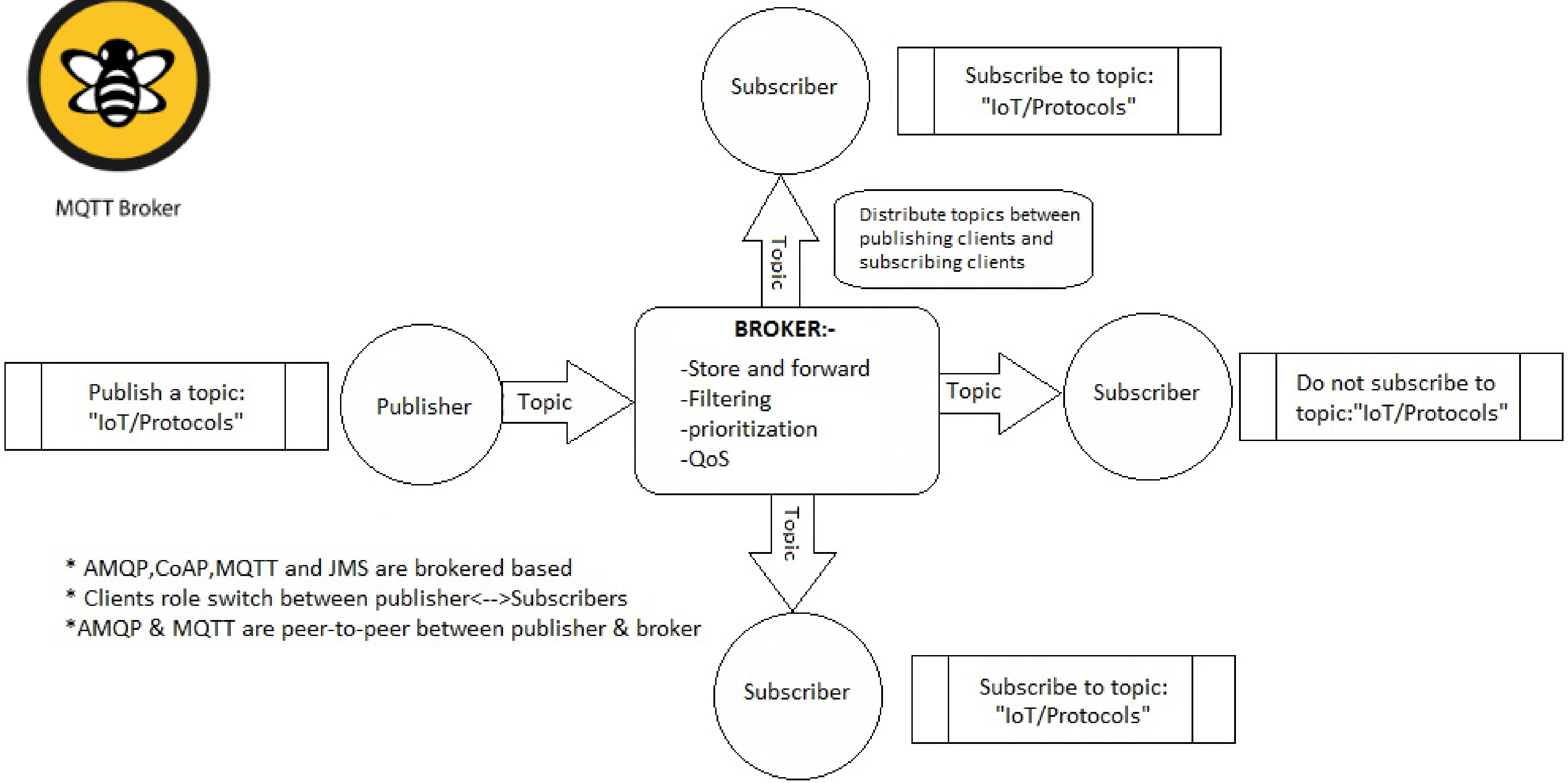
TOPIC	VALUE
$SYS/broker/version	mosquitto version 1.4.10
$SYS/broker/timestamp	Fri, 22 Dec 2017 08:19:25 +0000
$SYS/broker/uptime	2251623 seconds
$SYS/broker/clients/total	2
$SYS/broker/clients/inactive	0
$SYS/broker/clients/disconnected	0
$SYS/broker/clients/active	2
$SYS/broker/clients/connected	2
$SYS/broker/clients/expired	0
$SYS/broker/clients/maximum	3
$SYS/broker/messages/stored	61
$SYS/broker/messages/received	419773

[1] https://github.com/mqtt/mqtt.org/wiki/SYS-Topics
    ```

MQTT (Message Queuing Telemetry Transport)

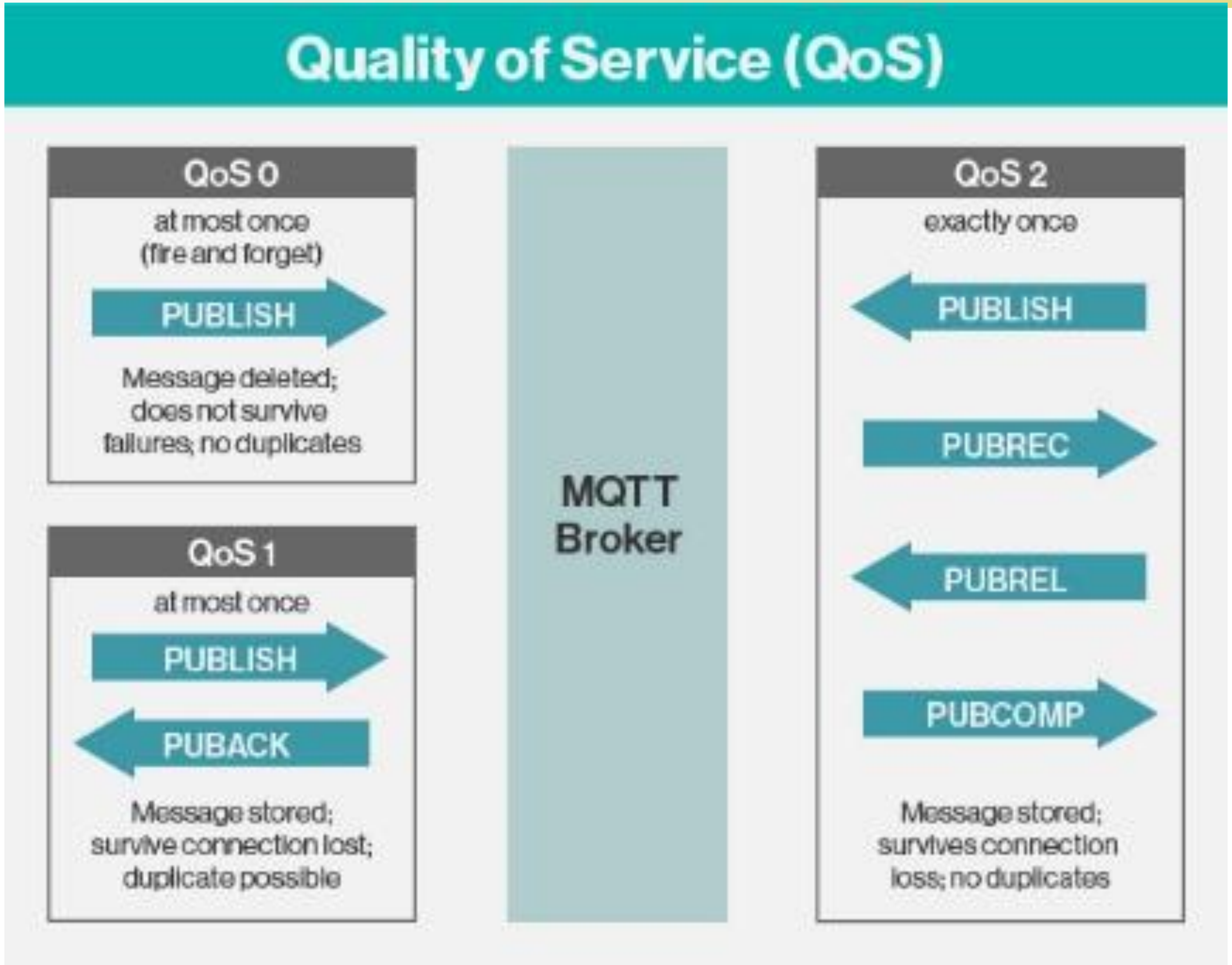
Open Connectivity for Mobile, M2M and IoT





* AMQP,CoAP,MQTT and JMS are brokered based
* Clients role switch between publisher<-->Subscribers
*AMQP & MQTT are peer-to-peer between publisher & broker

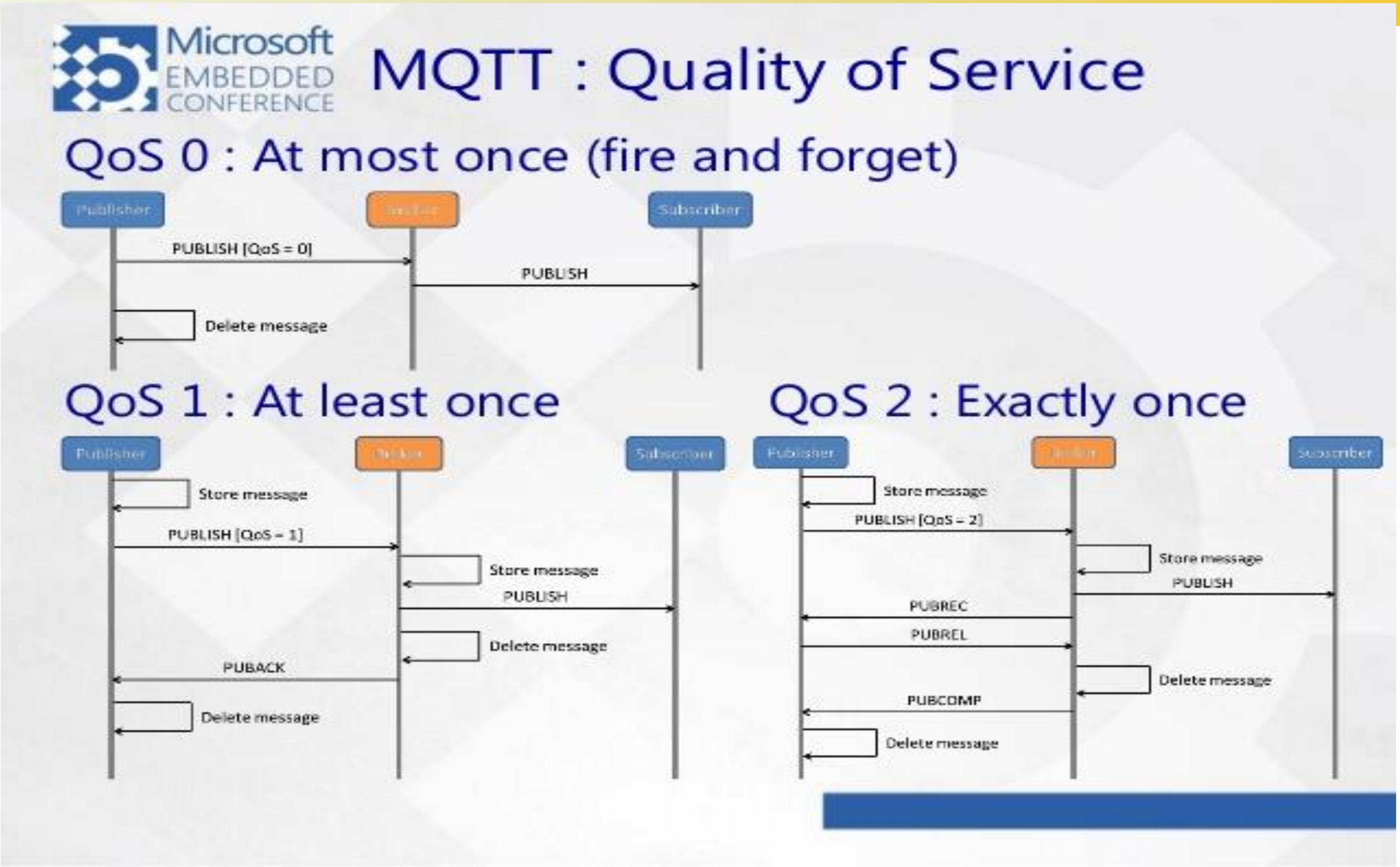
Quality of Service



- Higher QoS produces a throughput reduction in the order of 40%.
- Higher QoS slows down the message transmission (higher latency) by approximately 75%

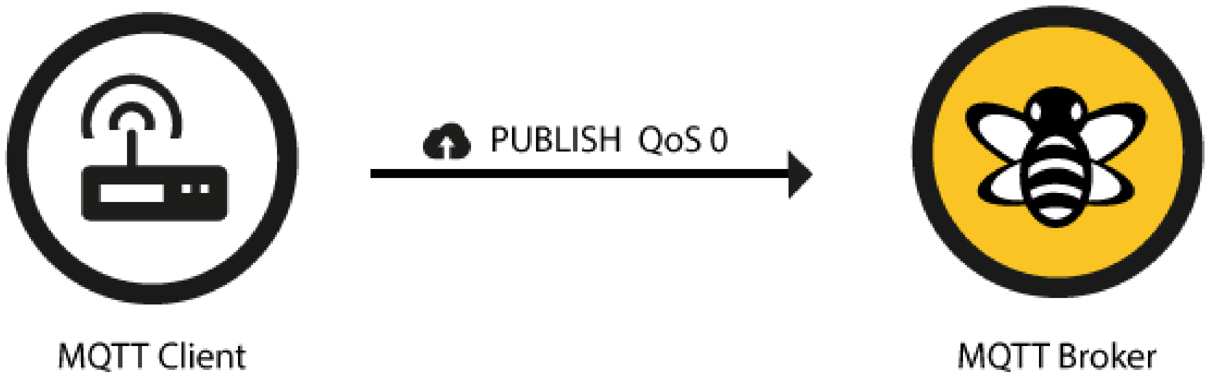
MQTT (Message Queuing Telemetry Transport)

Quality of Service



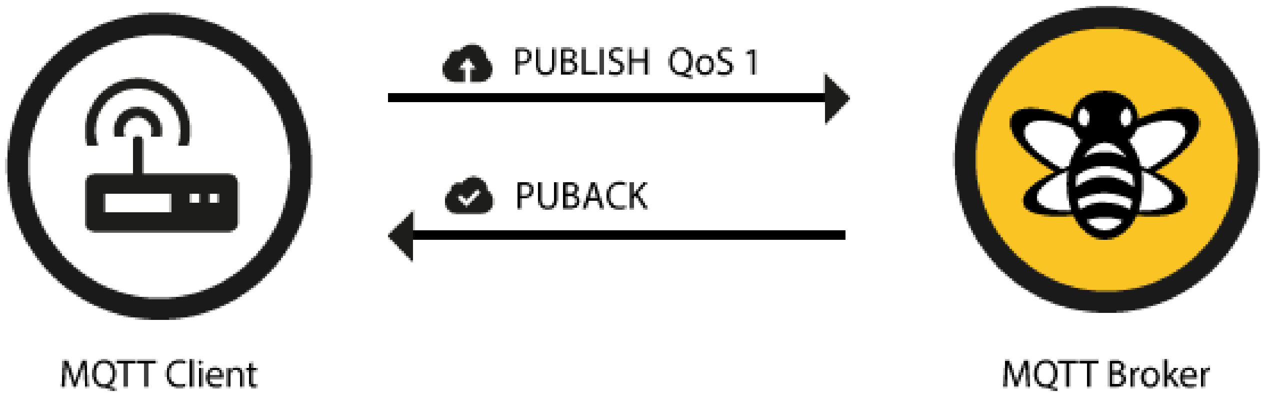
Publishing “QoS” (0-Unreliability)

- 0 – unreliable (aka “at most once”)
  OK for continuous streams, least overhead (1 message)
  “Fire and forget”
  TCP will still provide reliability
- Best-effort delivery.
- No guarantee of delivery.
- Recipient does not acknowledge receipt of the message.
- The message is not stored and retransmitted by the sender.



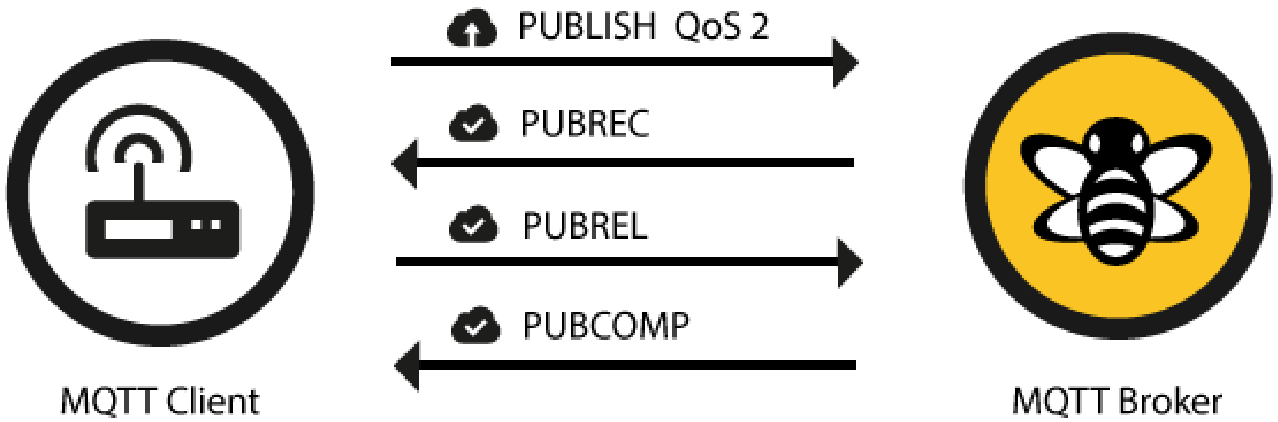
Publishing "QoS" (1-Reliability)

- 1 – delivery "at least once" (duplicates possible)
  - Used for alarms – more overhead (2 messages)
  - Contains message ID (to match with ACKed message)
- It guarantees that a message is delivered at least one time to the receiver.
- The sender stores the message until it gets a packet from the receiver that acknowledges receipt of the message.
- Message can be sent or delivered multiple times.



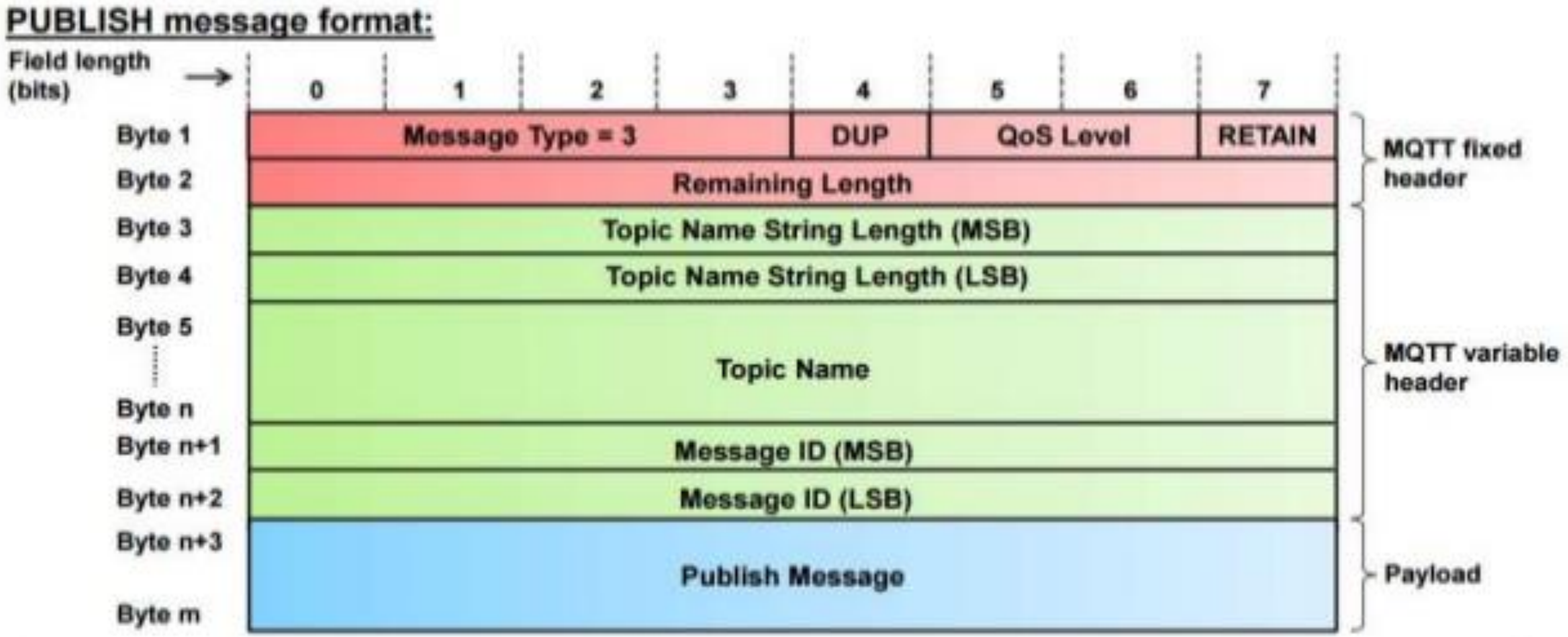
Publishing “QoS” (2-Reliability)

- 2 – delivery “exactly once”
  - Utmost reliability is important – most overhead (4 messages) and slowest
- It guarantees that each message is received only once by the intended recipients.
- Safest and slowest QoS.
- Guarantee is provided by at least two request/response flows (a four-part handshake) between the sender and the receiver



Message Format

PUBLISH	3	Publish message



PUBLISH message field	Description / Values
Topic Name with Topic Name String Length	Name of topic to which the message is published. The first 2 bytes of the topic name field indicate the topic name string length.
Message ID	A message ID is present if QoS is 1 (At-least-once delivery, acknowledged delivery) or 2 (Exactly-once delivery).
Publish Message	Message as an array of bytes. The structure of the publish message is application-specific.

MQTT (Message Queuing Telemetry Transport)

Message Types

Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

MQTT (Message Queuing Telemetry Transport)

Message Types

Message fixed header field	Description / Values	
Message Type	0: Reserved	8: SUBSCRIBE
	1: CONNECT	9: SUBACK
	2: CONNACK	10: UNSUBSCRIBE
	3: PUBLISH	11: UNSUBACK
	4: PUBACK	12: PINGREQ
	5: PUBREC	13: PINGRESP
	6: PUBREL	14: DISCONNECT
	7: PUBCOMP	15: Reserved
DUP	Duplicate message flag. Indicates to the receiver that this message may have already been received. 1: Client or server (broker) re-delivers a PUBLISH, PUBREL, SUBSCRIBE or UNSUBSCRIBE message (duplicate message).	
QoS Level	Indicates the level of delivery assurance of a PUBLISH message. 0: At-most-once delivery, no guarantees, «Fire and Forget». 1: At-least-once delivery, acknowledged delivery. 2: Exactly-once delivery. Further details see MQTT QoS.	
RETAIN	1: Instructs the server to retain the last received PUBLISH message and deliver it as a first message to new subscriptions. Further details see RETAIN (keep last message).	
Remaining Length	Indicates the number of remaining bytes in the message, i.e. the length of the (optional) variable length header and (optional) payload. Further details see Remaining length (RL).	

Publish Message

MQTT Publish Message

Fixed Header
- DF: Duplication Flag
- QoS = 0, 1, 2
- R: Retain

Topic Name (Variable Header)
- The length depends on the string chosen by the application
- It is likely, that to avoid collisions, users will give relatively long names such as:
  - `com/acme/myapp/mytopic`

Message ID
- Only present for QoS=1,2 and used to uniquely identify messages

Payload
- Contains the data, but no information on the serialisation format

```
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---+---+---+---+---+---+---+---+
| message type  | DF|   QoS | R |
+---------------+---+-------+---+
~  Remaining Length (1-4 bytes) ~
+-------------------------------+
|   Topic Name Length MSB       |
+---+---+---+---+---+---+---+---+
|   Topic Name Length LSB       |
+---+---+---+---+---+---+---+---+
|                               |
~          Topic Name           ~
|                               |
+---+---+---+---+---+---+---+---+
|        Message ID MSB         |
+---+---+---+---+---+---+---+---+
|        Message ID LSB         |
+---+---+---+---+---+---+---+---+
|                               |
~          Payload              ~
|                               |
+---+---+---+---+---+---+---+---+
```

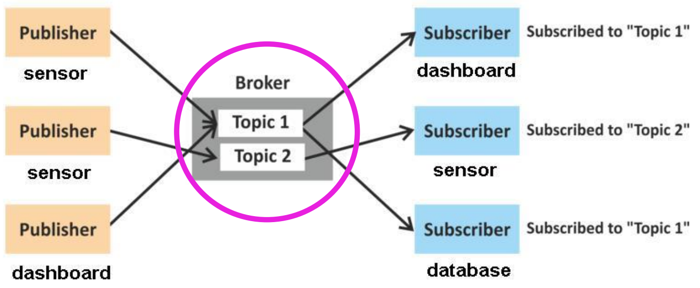
PRISMTECH

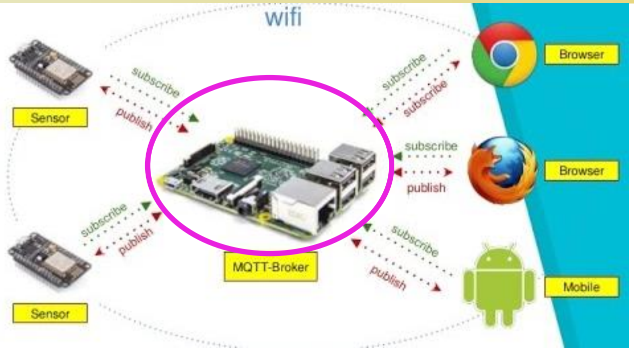
Topic Name

20 Character MQTT Topic Name

Proto	QoS	Data Message Size (bytes)	IP (v4) Headers	Total Overhead (bytes)
MQTT	0	(2 to 5) + 2 + 20 + length(Payload)	**IP**: 20-40 **TCP**: 20-40	**min** = 20 + 20 + 24 = **64** [TCP/IP] **max** = 40 + 40 + 27 = **107** [TCP/IP]
MQTT	1,2	(2 to 5) + 2 + 20 + 2 length(Payload)	**IP**: 20-40 **TCP**: 20-40	**min** = 20 + 20 + 26 = **66**[TCP/IP] **max** = 40 + 40 + 29 = **109** [TCP/IP]
DDS	DestinationOrder = DestinationTimestamp	44 + length(Payload)	**IP**: 20-40 **UDP**: 8 **TCP**: 20-40	**min** = 20 + 8 + 44 = **72** [UDP/IP] **max** = 40 + 8 + 44 = **94** [UDP/IP] **min** = 20 + 20 + 44 = **84** [TCP/IP] **max** = 40 + 40+ 44 = **124** [TCP/IP]
DDS	DestinationOrder = SourceTimestamp	56 + length(Payload)	**IP**: 20-40 **UDP**: 8 **TCP**: 20-40	**min** = 20 + 8 + 56 = **84** [UDP/IP] **max** = 40 + 8 + 56 = **106** [UDP/IP] **min** = 20 + 20 + 56 = **96** [TCP/IP] **max** = 40 + 40+ 56 = **136** [TCP/IP]

Implementation





https://mosquitto.org/

1999 Invented by Dr. Andy Stanford-Clark (IBM), Arlen Nipper (now Cirrus Link Solutions)

2004 MQTT.org open community

2011 - Eclipse PAHO MQTT open source project

2013 – MQTT Technical Committee formed

Cimetrics, Cisco, Eclipse, dc-Square, Eurotech, IBM, INETCO Landis & Gyr, LSI, Kaazing, M2Mi, Red Hat, Solace, Telit Comms, Software AG, TIBCO, WSO2

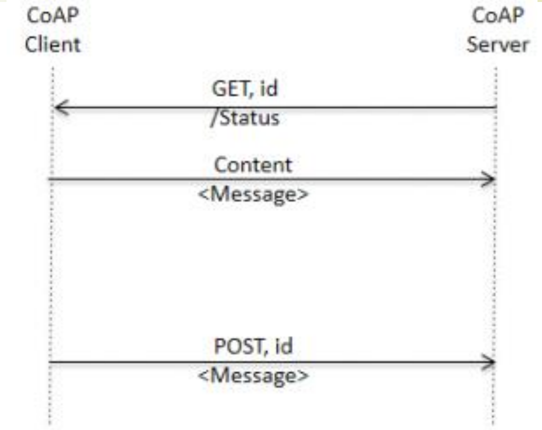
Evolution of an open technology

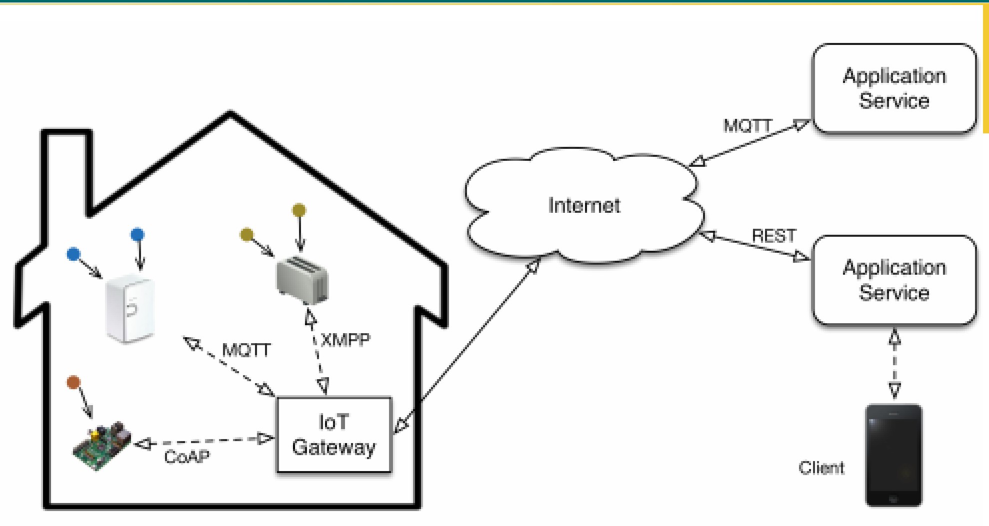
Public MQTT Brokers

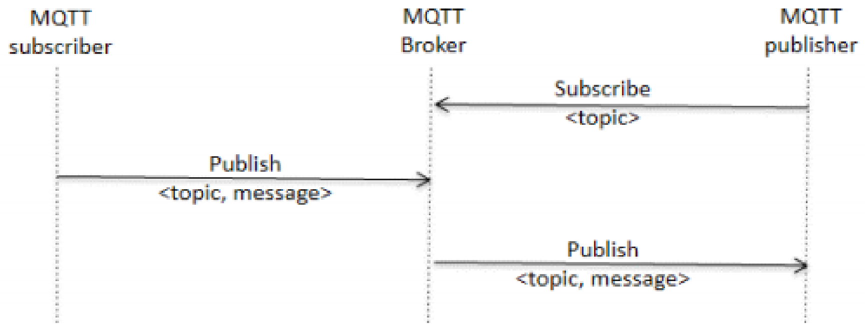
- https://test.mosquitto.org/
- https://iot.eclipse.org/getting-started/#sandboxes
- https://www.hivemq.com/public-mqtt-broker/
- https://www.cloudmqtt.com/
- https://netpie.io/

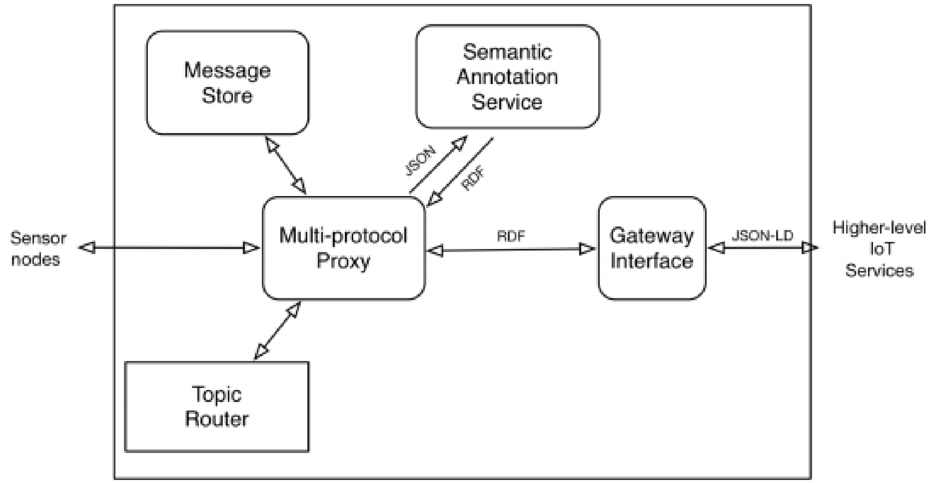
Benchmark Tool
https://github.com/krylovsk/mqtt-benchmark

Message Transfer (1/3)

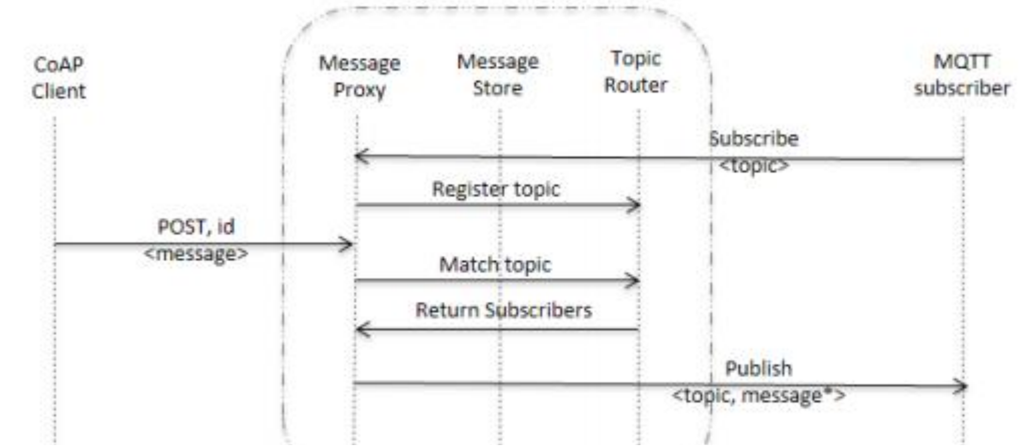


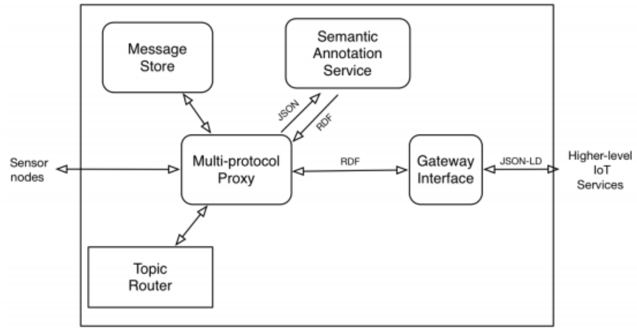


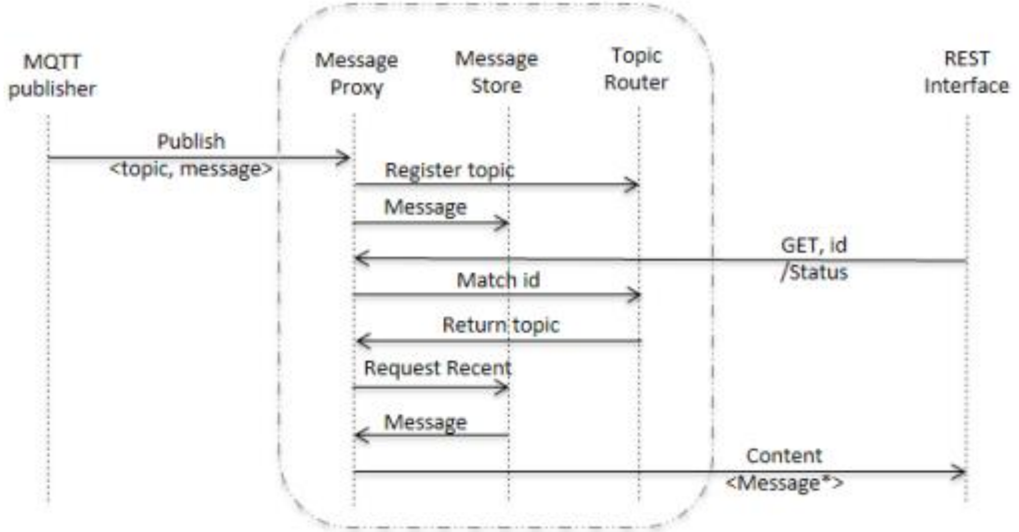


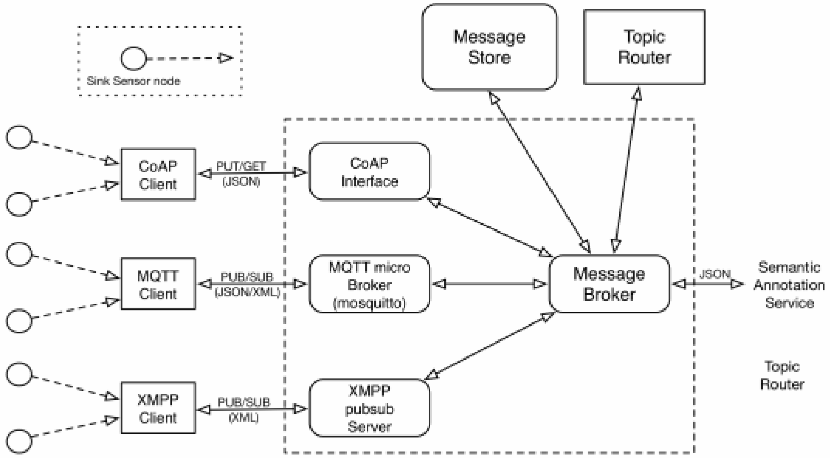


Message Transfer (2/3)



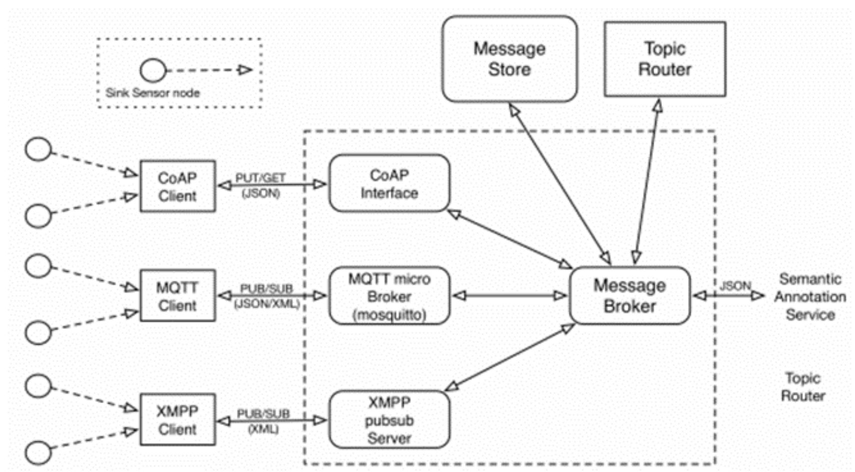


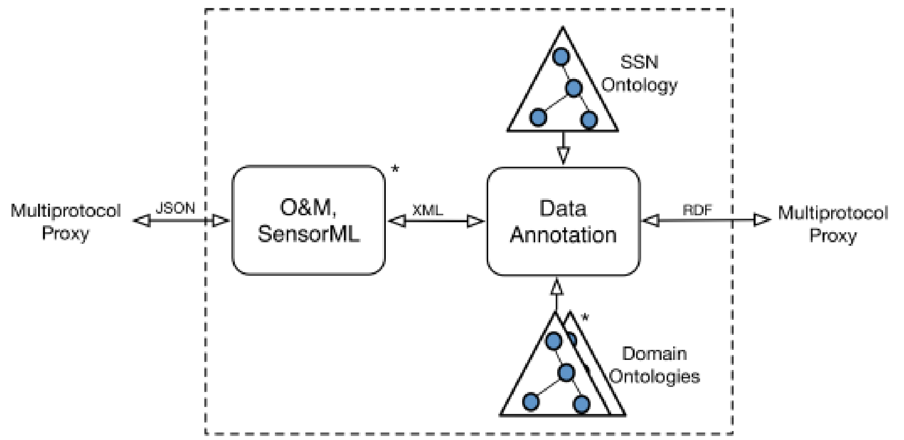




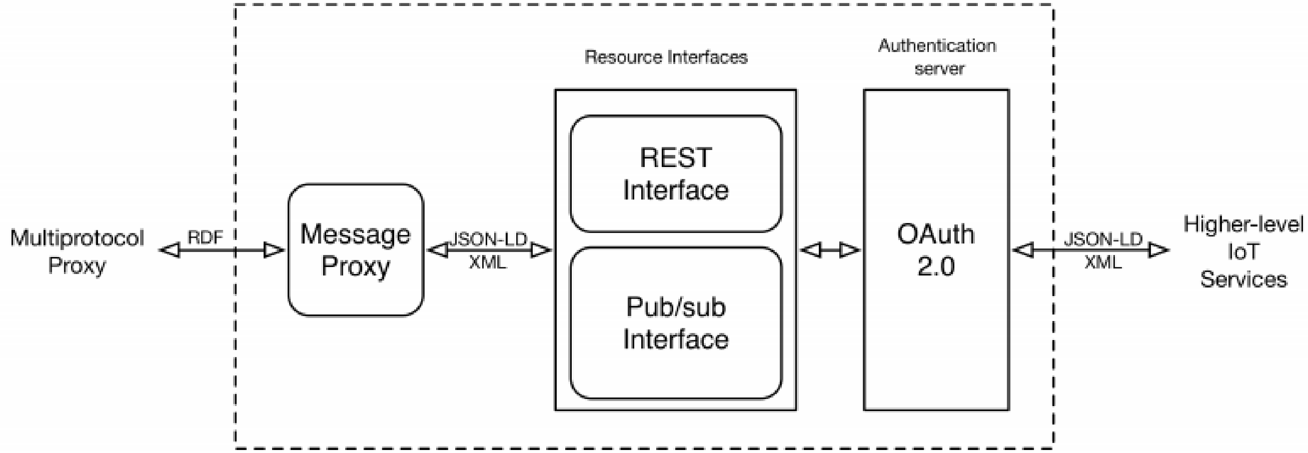
Implementation Examples with CoAP and MQTT

Message Transfer (3/3)





Semantic Annotation Service



Gateway Service

A Comparison between HTTP, CoAP and MQTT

Basis of	HTTP (HyperText Transfer Protocol)	COAP (Constrained Application Protocol)	MQTT (Message Query Telemetry Transport)
Architecture	Client/Server	Client/Server or Client/Broker	Client/Broker
Communication Type	Request-Response Model	Request-Response model. One-to-one communication	Publish-Subscribe model. Many-to-many communication
Messaging Mode	Asynchronous	Asynchronous/Synchronous.	Asynchronous
Semantics/Methods	Get, Post, Head, Put, Patch, Options, Connect, Delete	Get, Post, Put, Delte	Connect, Disconnect, Publish, Subscribe, Unsubscribe, Close
Transport layer protocol: Port - QoS	TCP: 80/443 QoS Limited via TCP	UDP: 5683/5684 Confirmable/non-confirmable message	TCP:1883/8883 QoS 0/1/2
Header size	Undefined	It has 4 bytes sized header	It has 2 bytes sized header
Message Size	Large and Undefined (depends on the web server or the programming technology)	Small and Undefined (normally small to fit in single IP datagram)	Small and Undefined (up to 256 MB maximum size)
RESTful based		Yes it uses REST principles	No it does not uses REST principles
Persistence support	Upon HTTP version	It does not has such support	It supports and best used for live data communication
Message Labelling		It provides by adding labels to the messages.	It has no such feature.
Usability/Security	TLS/SSL	DTLS/IPSEC It is used in Utility area networks and has secured mechanism	TLS/SSL It is used in IoT applications and is secure

CoAP/MQTT/AMQP/HTTP-REST/XMPP

Key performance indicator	Most promising protocol				Least promising protocol
Latency					
Over a LAN	CoAP	MQTT QoS 0	AMQP	HTTP/REST	XMPP
Over a mobile network	MQTT QoS 0	CoAP	WebSocket	MQTT QoS 1	
Bandwidth consumption	CoAP	MQTT	AMQP and XMPP	DDS	HTTP/REST
Throughput	MQTT	DDS	CoAP	AMQP	XMPP
Reliability	MQTT	AMQP	CoAP	HTTP/REST	
Energy consumption	CoAP	MQTT	AMQP	HTTP/REST	
Developers' preference in recent IoT applications	MQTT	HTTP/REST	WebSocket	HTTP 2.0	CoAP, AMQP, XMPP, DDS
Researchers' preference in IoT agriculture applications	MQTT	HTTP/REST	CoAP		

References

[VB17] Ovidiu Vermesan and Joel Bacquet, "**Cognitive Hyperconnected Digital Transformation: Internet of Things Intelligence Evolution**", River Publishers, 2017.

[BHC+18] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "**The Industrial Internet of Things (IIoT): An Analysis Framework**", Computers in Industry 101, October 2018.

[BS19] Rajkumar Buyya and Satish Narayana Srirama, "**Fog and Edge Computing: Principles and Paradigms**", Wiley Series on Parallel and Distributed Computing, 2019.

[RS19] Ammar Rayes and Samer Salam, "**Internet of Things from Hype to Reality: The Road to Digitization**", 2nd Edition, Springer, 2019.

[LEA20] Perry Lea, "**IoT and Edge Computing for Architects: Implementing edge and IoT systems from sensors to clouds with communication systems, analytics, and security"**, 2nd Edition, Packt Publishing, 2020.

AMQP 1.0: https://www.youtube.com/watch?v=ODpeIdUdClc&list=PLmE4bZU0qx-wAP02i0I7PJWvDWoCytEjD&index=2
https://www.youtube.com/watch?v=g3e9lDlMn5M

XMPP: https://www.youtube.com/watch?v=92egt5-UDwo
https://www.youtube.com/watch?v=fz0yDNwEydU

DDS: https://www.youtube.com/watch?v=7IV49wKxs4c (Security Issues)
https://www.youtube.com/watch?v=k1P1cQUZl-A (Demo)
https://www.youtube.com/watch?v=u-saogMmKOo

ITCS447
Lecture 9
ESP32 Web Server

Asst. Prof. Dr. Thitinan Tantidham

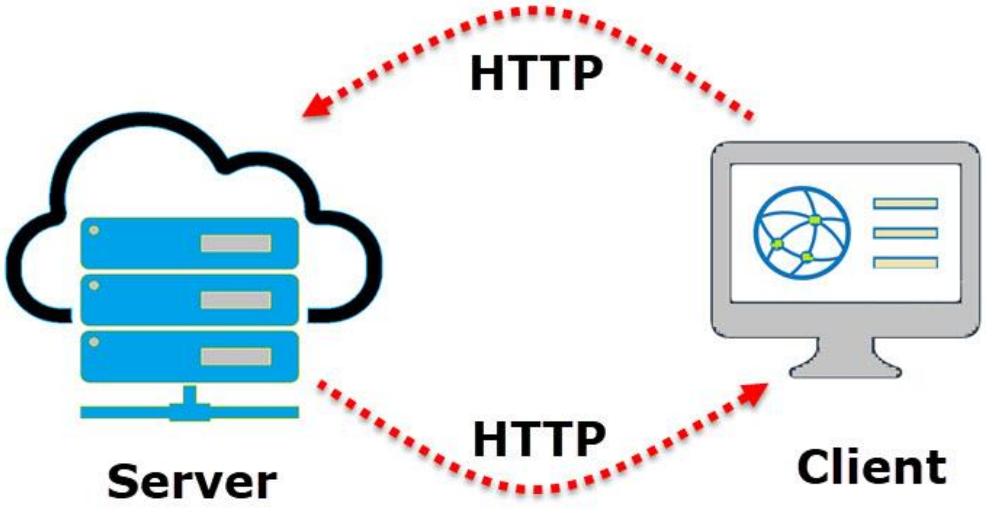
มหาวิทยาลัยมหิดล
Mahidol University

Content

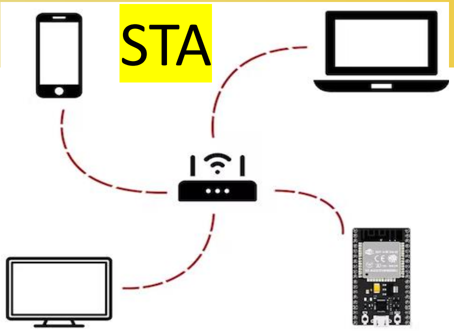
- ESP32 Mode
- ESP32 Webserver
- ESP32 REST API
- Serial Peripheral Interface Flash File System (SPIFFS)

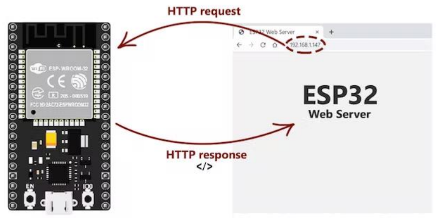
Web Server

- HTTP 1.1/2.0/3.0
- HTTP status code
  - When a request from a web browser is sent to a web server, the server returns a code as a response, for examples:
    - 200: means the connection is established correctly,
    - 404: indicates that the address is not correct).
- Web Page: HTML, XML, CSS, JS …

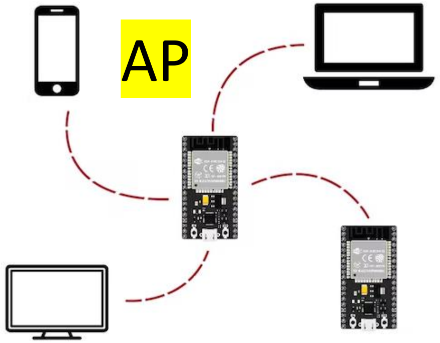


ESP32 Mode





- ESP32 as a Station Mode (STA)
  - ESP32 is connected to the Wi-Fi router as a Client and can access the Internet through the router.
- ESP32 as an Access Point Mode (AP)
  - ESP32 can act as a router and creates a local wifi network with the desired name and password.
  - The number of devices is limited. It’s also called Soft Access Point.
- Both modes can be programmed as a Webserver
  - HTTP/HTTPs
  - TCP Port 80/443



https://www.hackster.io/electropeak/create-a-web-server-w-esp32-tutorial-a9a392

ESP32 vs ESP8266 Web Server

```
#include <Arduino.h>
#ifdef ESP32
  #include <WiFi.h>
  #include <AsyncTCP.h>
#else
  #include <ESP8266WiFi.h>
  #include <ESPAsyncTCP.h>
#endif
#include <ESPAsyncWebServer.h>

AsyncWebServer server(80);

#include "config.h"
```

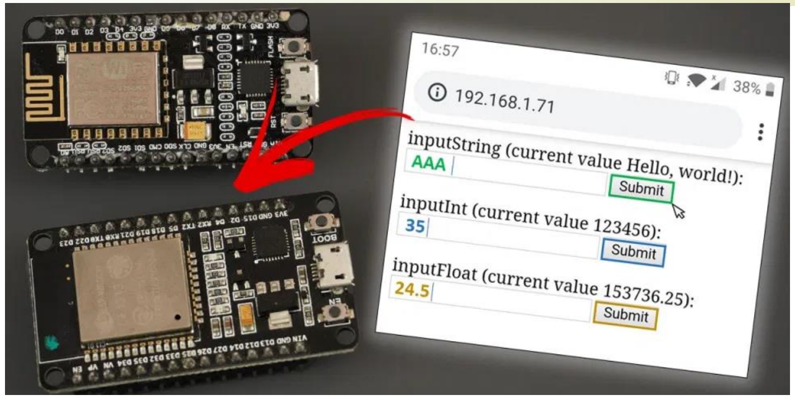
```
String HTML = "<!DOCTYPE html>\
<html>\
<body>\
<h1>My First Web Server with ESP32 -
Station Mode &#128522;</h1>\
</body>\
</html>";

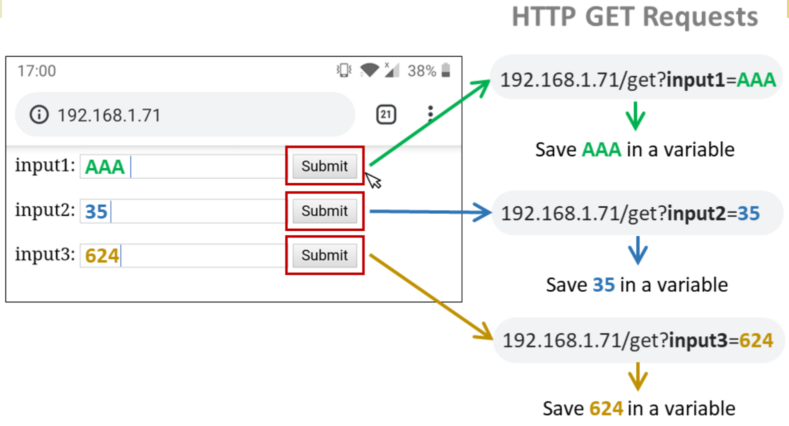
 server.send(200, "text/html", HTML);
```

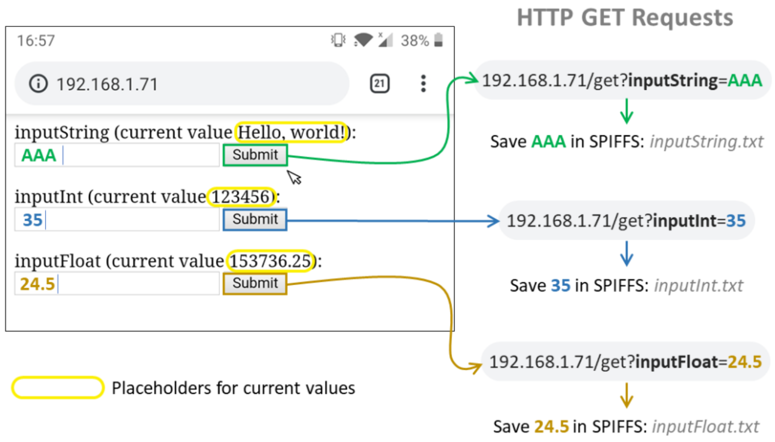
```
client.println("<!DOCTYPE html><html>");
client.println("<head><meta name=\"viewport\"
content=\"width=device-width, initial-scale=1\">");
client.println("<link rel=\"icon\" href=\"data:,\">");
```

ESP32 Web Server [RANDOMNERDTUTORIALS]

Example: Input Data



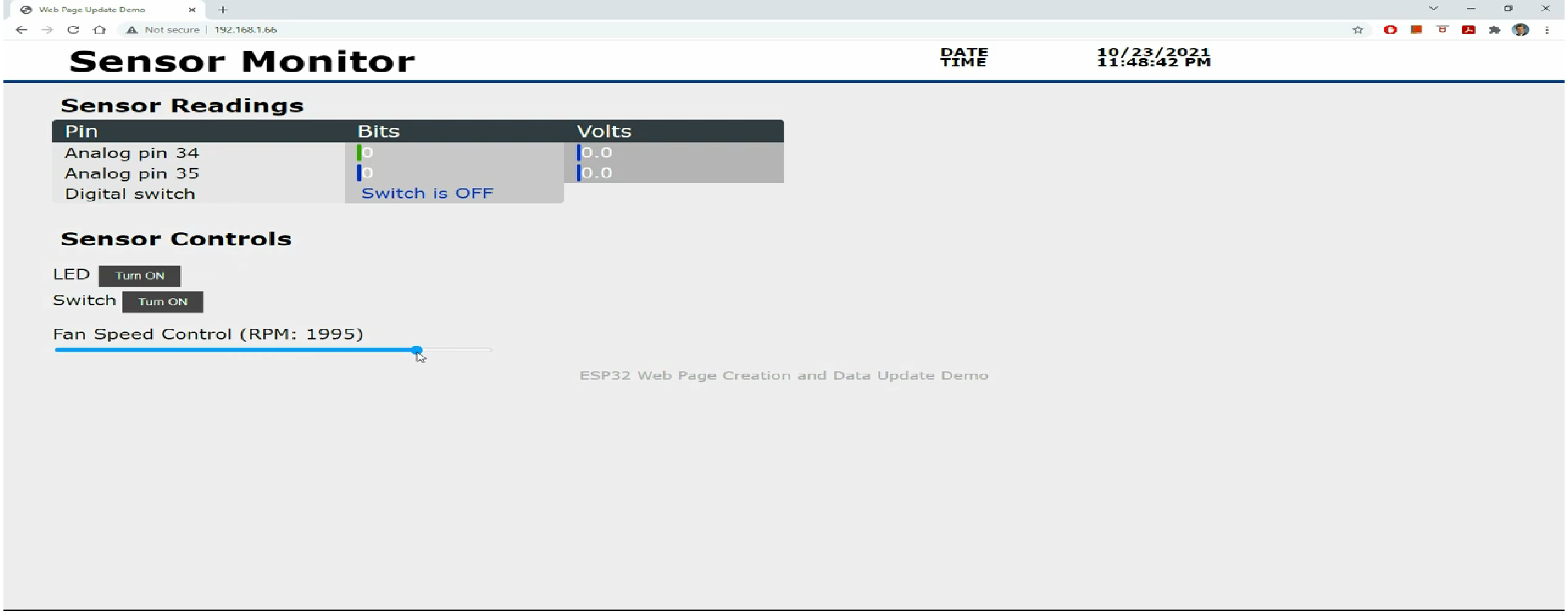




https://randomnerdtutorials.com/esp32-esp8266-input-data-html-form/

ESP32 Web Server (include other HTML Libraries)

Example: Page Creation and Content Updating



https://www.youtube.com/watch?v=pL3dhGtmcMY

https://github.com/KrisKasprzak/ESP32_WebPage

SPIFFS (Serial Peripheral Interface Flash File System)

- SPIFFS is a lightweight filesystem created for microcontrollers with a flash chip, which is connected by SPI bus.
- ESP32 contains a flash memory with SPIFFS.
- Setup and Test:
  - https://randomnerdtutorials.com/install-esp32-filesystem-uploader-arduino-ide/
- ESP32 Webserver using SPIFFS
  - https://randomnerdtutorials.com/esp32-web-server-spiffs-spi-flash-file-system/

https://randomnerdtutorials.com/install-esp32-filesystem-uploader-arduino-ide/

References

[VB17] Ovidiu Vermesan and Joel Bacquet, "**Cognitive Hyperconnected Digital Transformation: Internet of Things Intelligence Evolution**", River Publishers, 2017.

[BHC+18] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "**The Industrial Internet of Things (IIoT): An Analysis Framework**", Computers in Industry 101, October 2018.

[BS19] Rajkumar Buyya and Satish Narayana Srirama, "**Fog and Edge Computing: Principles and Paradigms**", Wiley Series on Parallel and Distributed Computing, 2019.

[RS19] Ammar Rayes and Samer Salam, "**Internet of Things from Hype to Reality: The Road to Digitization**", 2nd Edition, Springer, 2019.

[LEA20] Perry Lea, "**IoT and Edge Computing for Architects: Implementing edge and IoT systems from sensors to clouds with communication systems, analytics, and security"**, 2nd Edition, Packt Publishing, 2020.

HTTP Status Code:
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

https://www.electronicshub.org/esp32-web-server/

https://www.dfrobot.com/blog-1574.html

https://github.com/lorol/LITTLEFS

https://techtutorialsx.com/2019/04/07/esp32-https-web-server/

https://www.tutorialspoint.com/esp32_for_iot/esp32_for_iot_transmitting_data_over_wifi_using_https.htm

ITCS447
Lecture 10

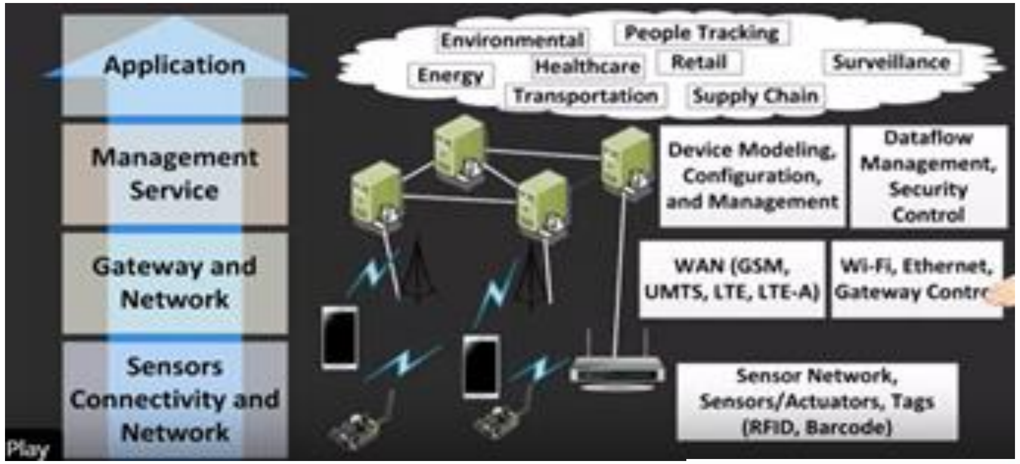
IoT Platforms and Infrastructures
Edge/Fog/Cloud Computing and Technologies

Asst. Prof. Dr. Thitinan Tantidham

Outline

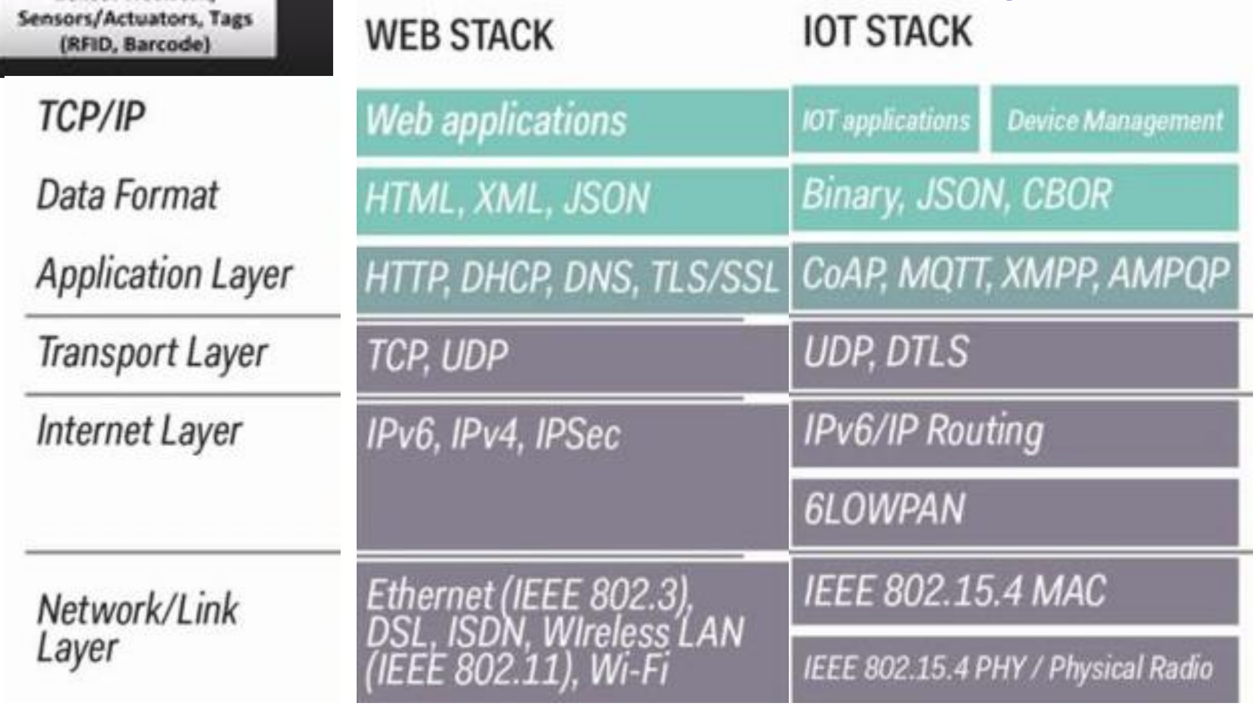
- Introduction: IoT Architecture Layers vs TCP/IP Layers
- Multi-Tier Computing
- Fog and Edge Computing
- Edge Computing

IoT Architecture Layers vs TCP/IP Layers: IoT & Web



IoT Architecture Layers

TCP/IP Protocol Stack Layers

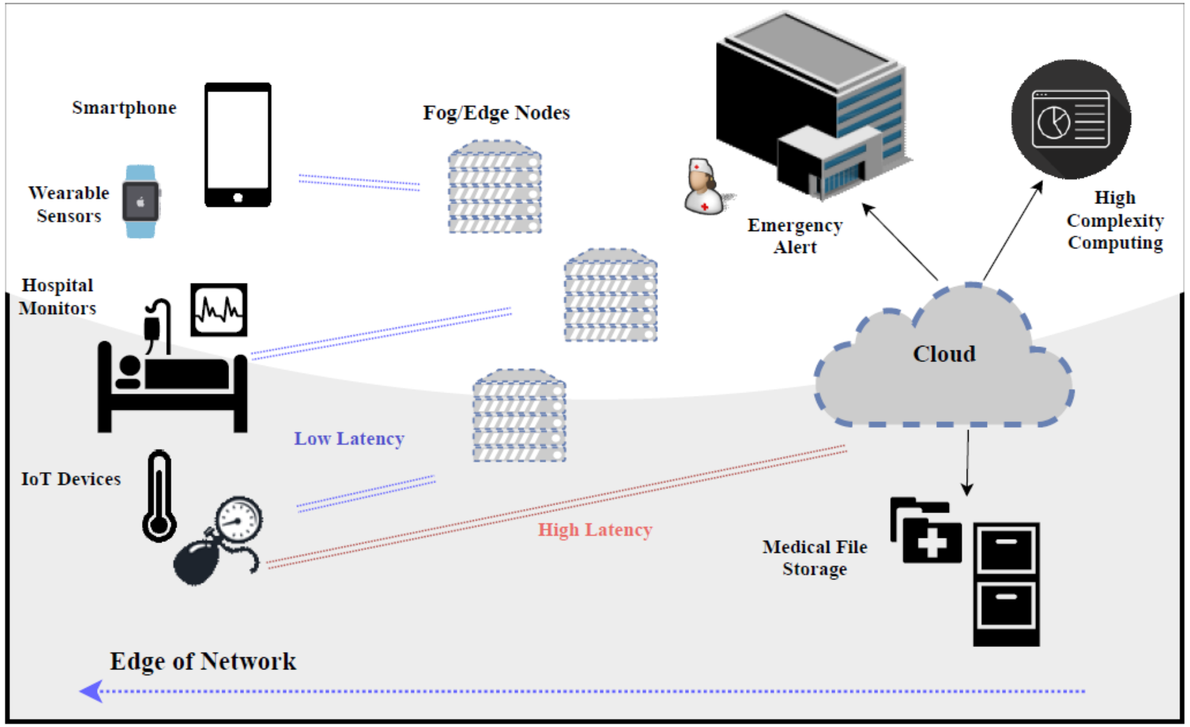


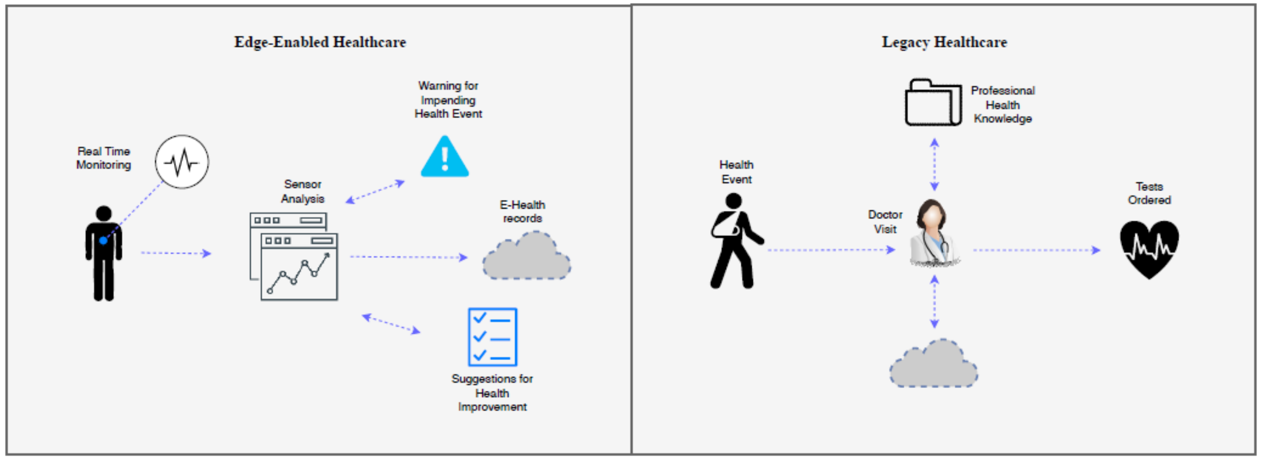
TCP/IP	WEB STACK	IOT STACK
TCP/IP	Web applications	IOT applications / Device Management
Data Format	HTML, XML, JSON	Binary, JSON, CBOR
Application Layer	HTTP, DHCP, DNS, TLS/SSL	CoAP, MQTT, XMPP, AMPQP
Transport Layer	TCP, UDP	UDP, DTLS
Internet Layer	IPv6, IPv4, IPSec	IPv6/IP Routing 6LOWPAN
Network/Link Layer	Ethernet (IEEE 802.3), DSL, ISDN, Wireless LAN (IEEE 802.11), Wi-Fi	IEEE 802.15.4 MAC IEEE 802.15.4 PHY / Physical Radio

Go to:
Coursera, Prof.Jong-Moon Chung Yonsei University – *0,a,b video files

Introduction

Comparison of Edge-Enabled and Legacy Healthcare





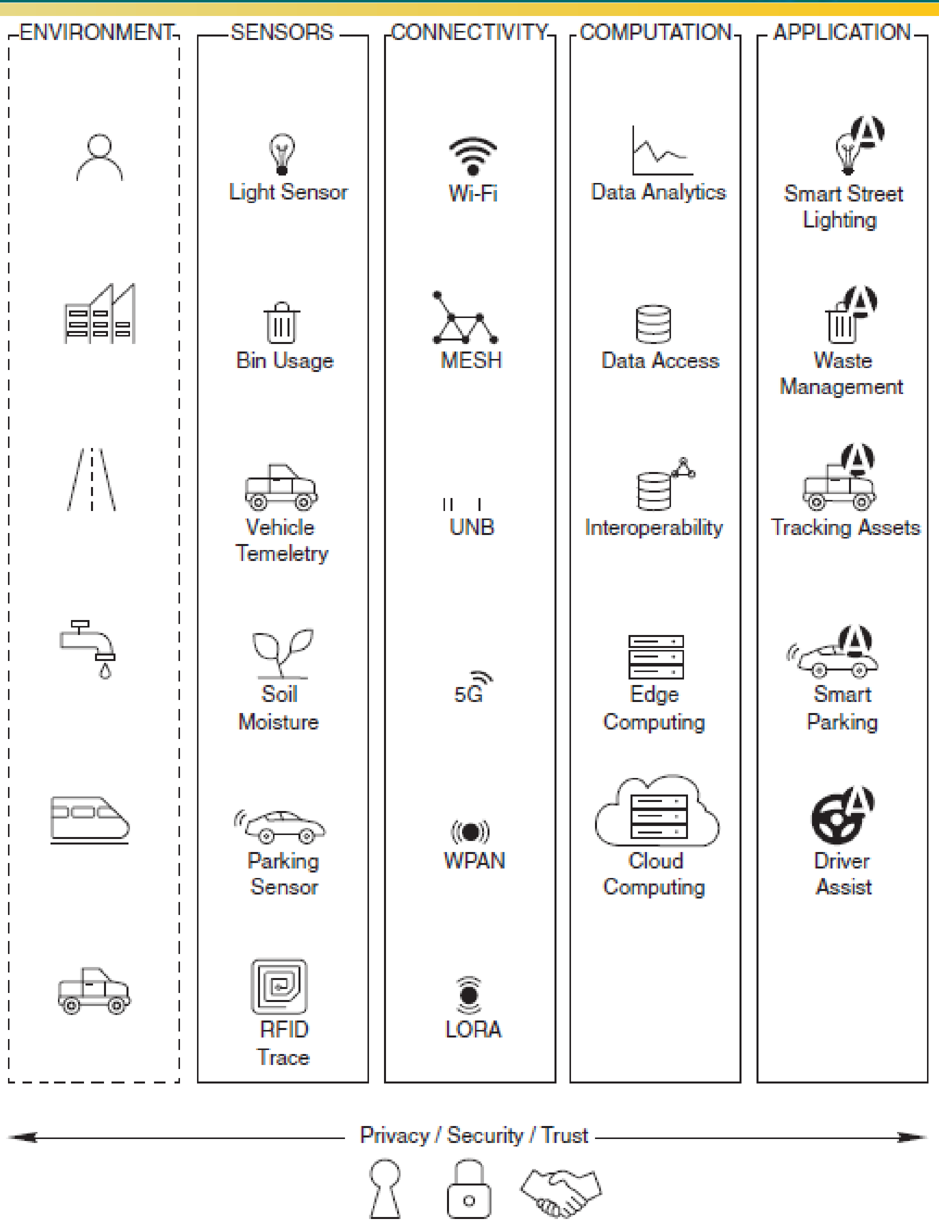
Ref: M. Hartmann, U. S. Hashmi, A. Imran, "Edge Computing in Smart Health Care Systems: Review, Challenges, and Research Directions, 2017

IoT Ecosystem [DF20]

- IoT Ecosystem consists of multiple components that allow business, governments, and consumers to connect their **inter**dependent IoT devices and technologies, such as a smart city environment.

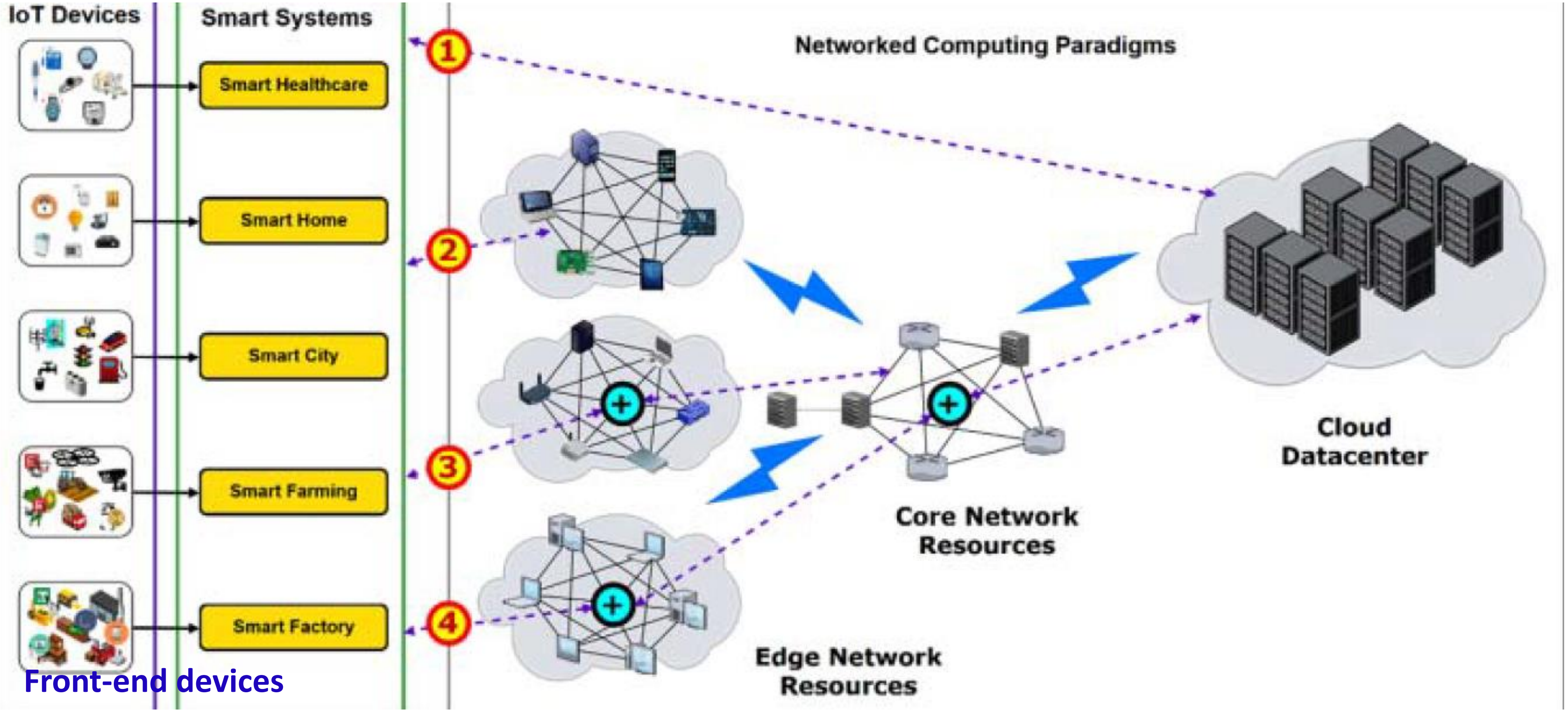
- Stakeholders may play one or more different roles. These include multiple providers and developers for end users such as sensors, connectivities, information, applications, analytics, services, and platforms.

  - ✓ Sensor/Actuators
  - ✓ Device connectivity
  - ✓ Application in the smart device
  - ✓ The Access Network and the Internet (Cellular: *G: 4G/5G, NBIoT/Sigfox/LoRaWAN)
  - ✓ The application (and processing) on the cloud
  - ✓ Data Analytics
  - ✓ Security



Introduction

IoT and Cloud-centric IoT



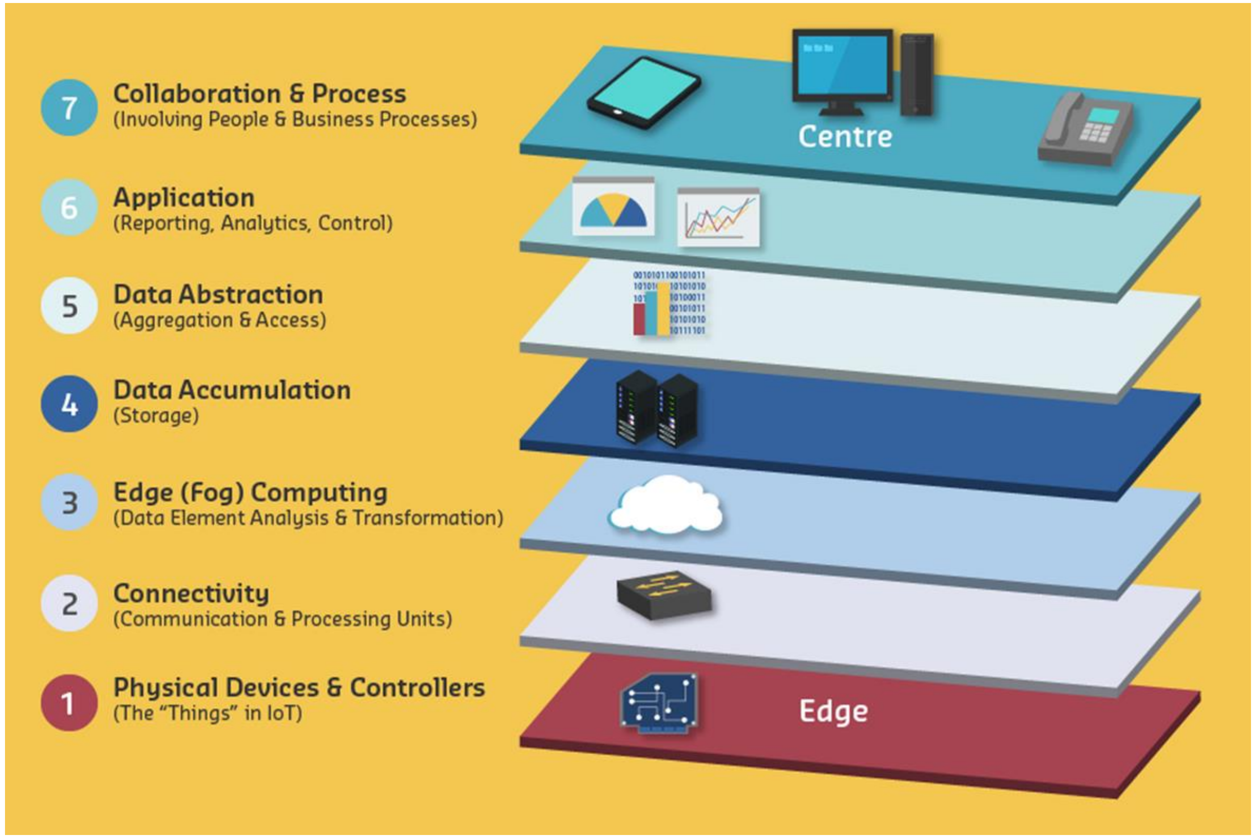
IoT and Cloud-centric IoT

- Commonly, an IoT system follows the architecture of the Cloud-centric Internet of Things (CIoT) in which the physical objects are represented in the form of Web resources that are managed by the servers in the global Internet.

- CIoT faces challenges in BLURS
  - **B**andwidth: the increasing large and high-frequent rate produced in objects in IoT will exceed the bandwidth availability, e.g. a self-driving vehicle can generate gigabytes of data per second due to the need for real-time video streaming.
  - **L**atency: The requirement of controlling the end-to-end latency within tens of milliseconds.
  - **U**ninterrupted: The long distance between cloud and the front-end IoT devices can face issues derived from the unstable and intermittent network connectivity.
  - **R**esource-constrained: CIoT systems usually require front-end devices to continuously stream their data to the cloud.
  - **S**ecurity: rarely updated software and less resource capacity of front-end devices. The attacker may also damage or control the front-end device and send false data to the cloud.

IoT Reference Model

The Edge (Fog) of the Cloud is becoming a global platform for the computation and interaction between machines and smart objects, in real-time applications.



Example 1

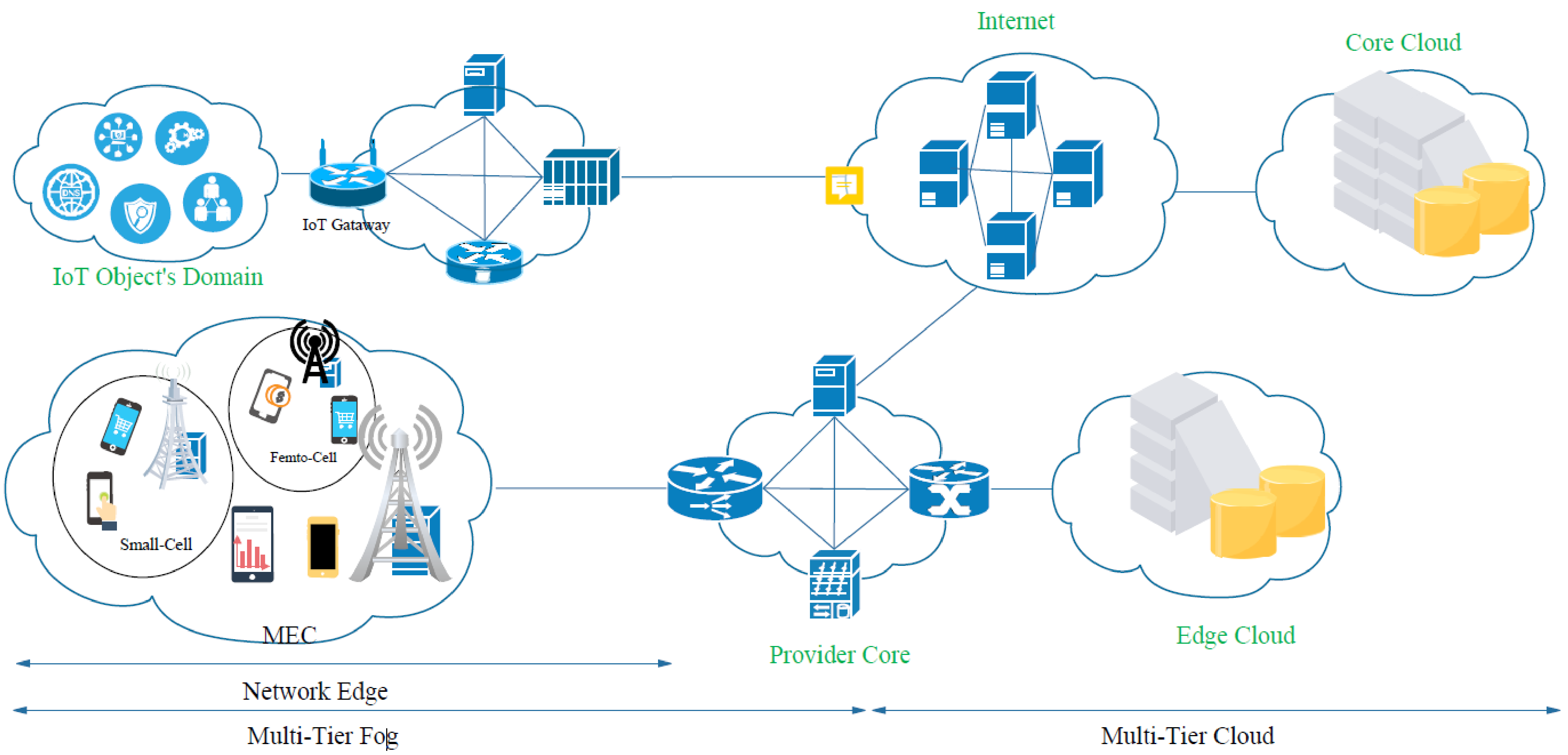
Upper Tier:
Cloud Computing

Immediate Tier:
Edge Computing

Lower Tier:
End Devices

MEC: Multi-access or Mobile Edge Computing

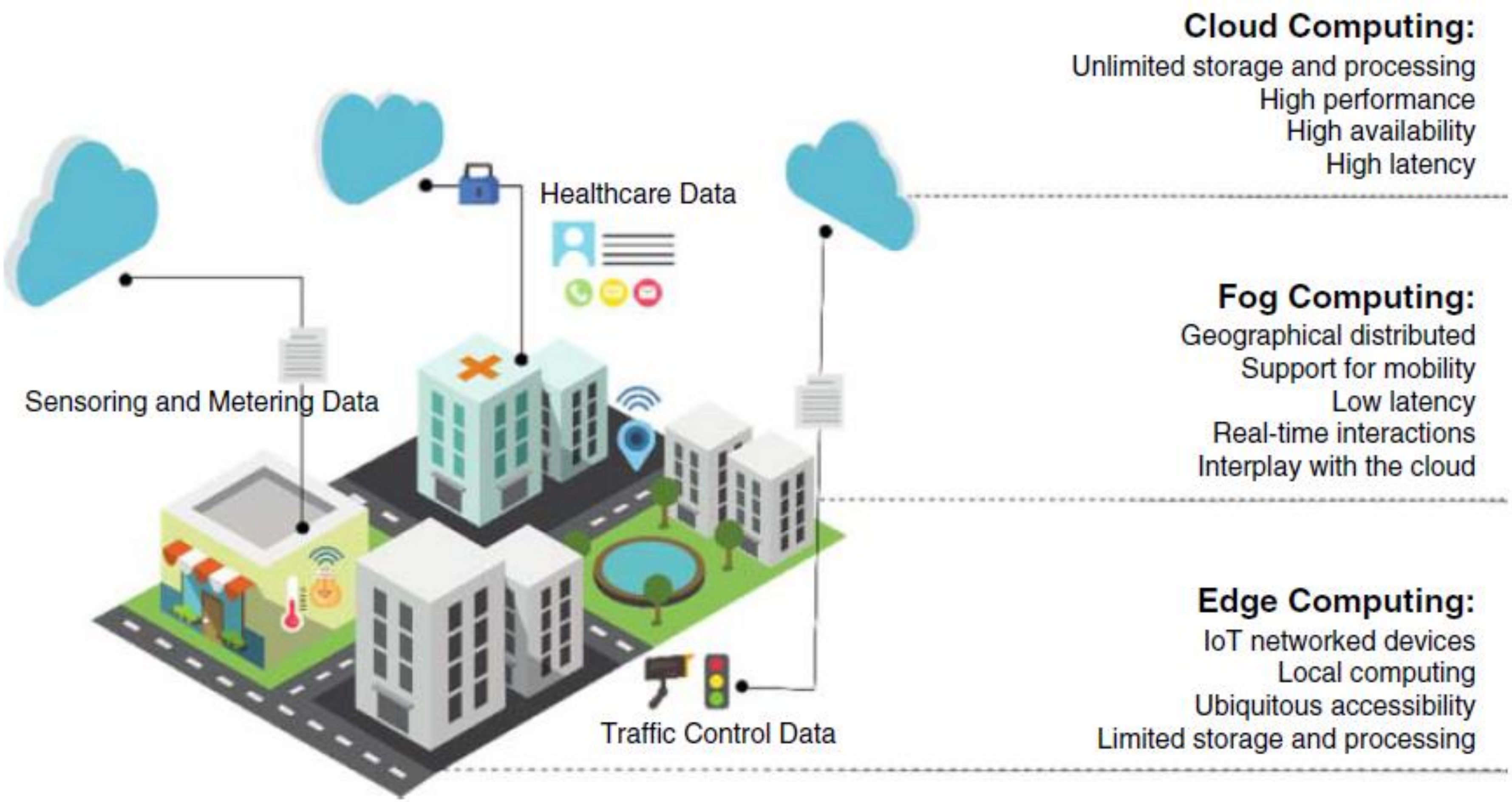
Example 2



Internet
Core Cloud
IoT Gataway
IoT Object's Domain
Femto-Cell
Small-Cell
MEC
Provider Core
Edge Cloud
Network Edge
Multi-Tier Fog
Multi-Tier Cloud

Multi-Tier Computing

Example 3



Multi-Tier Computing

Example 4

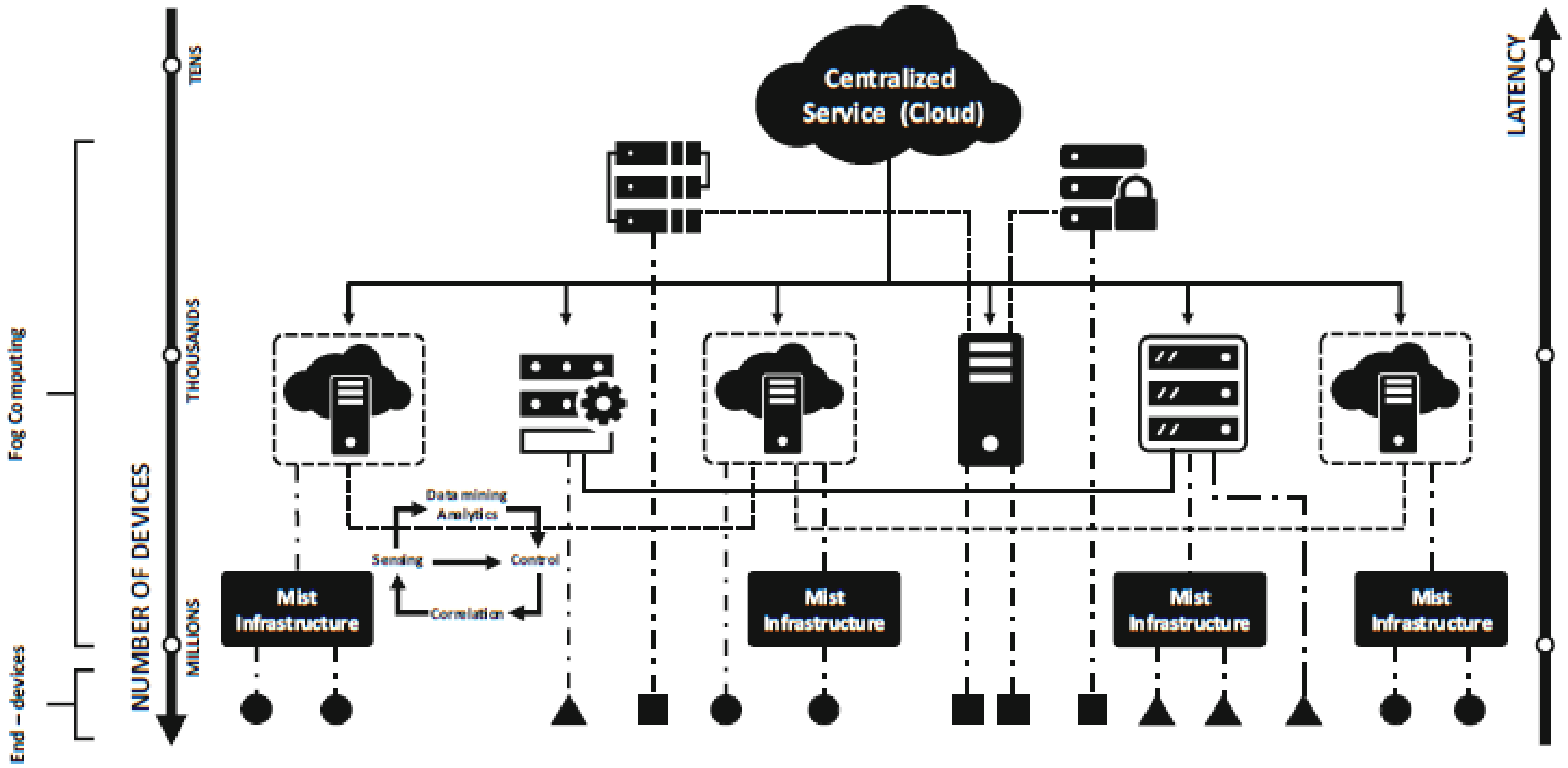
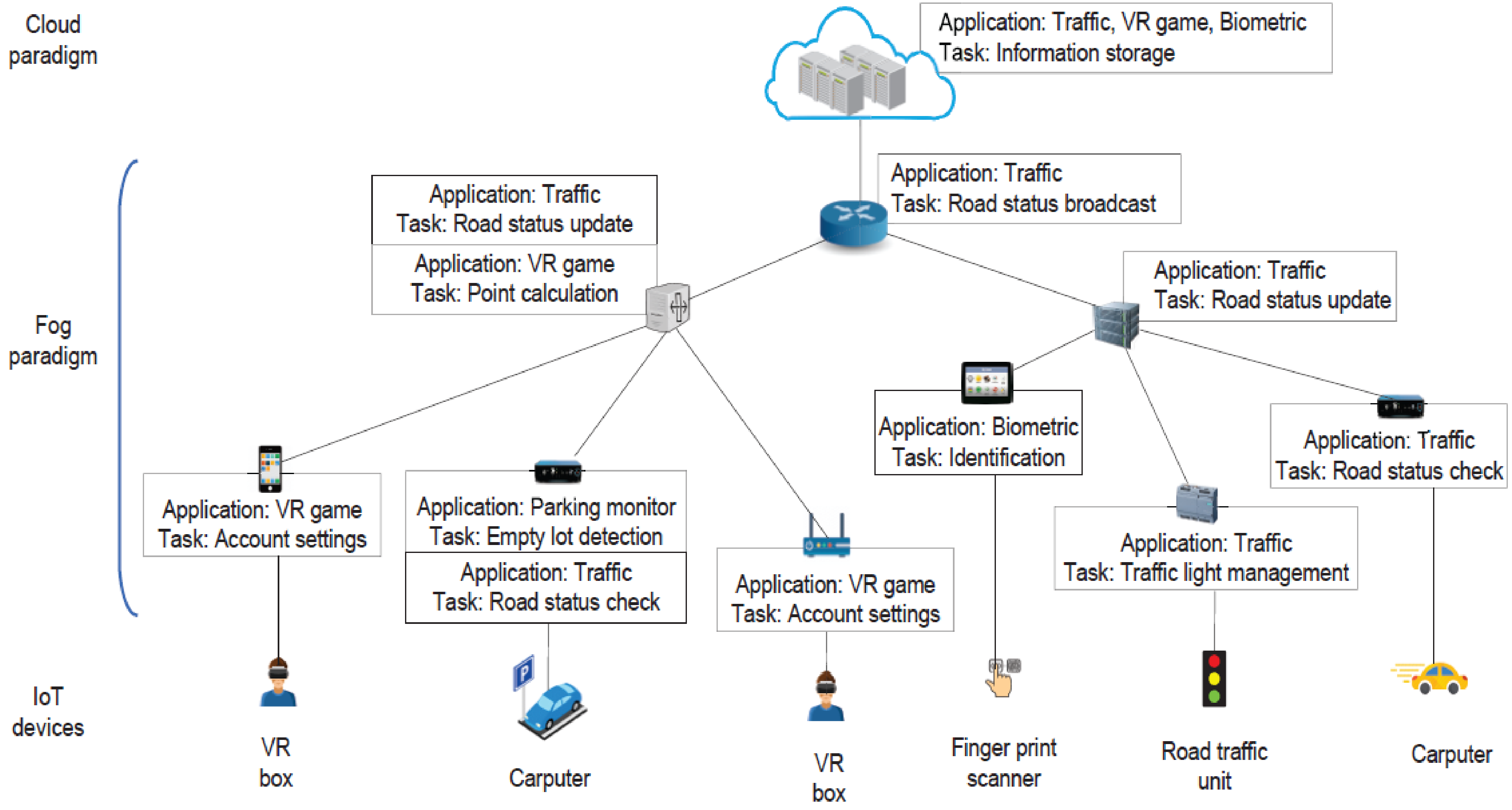
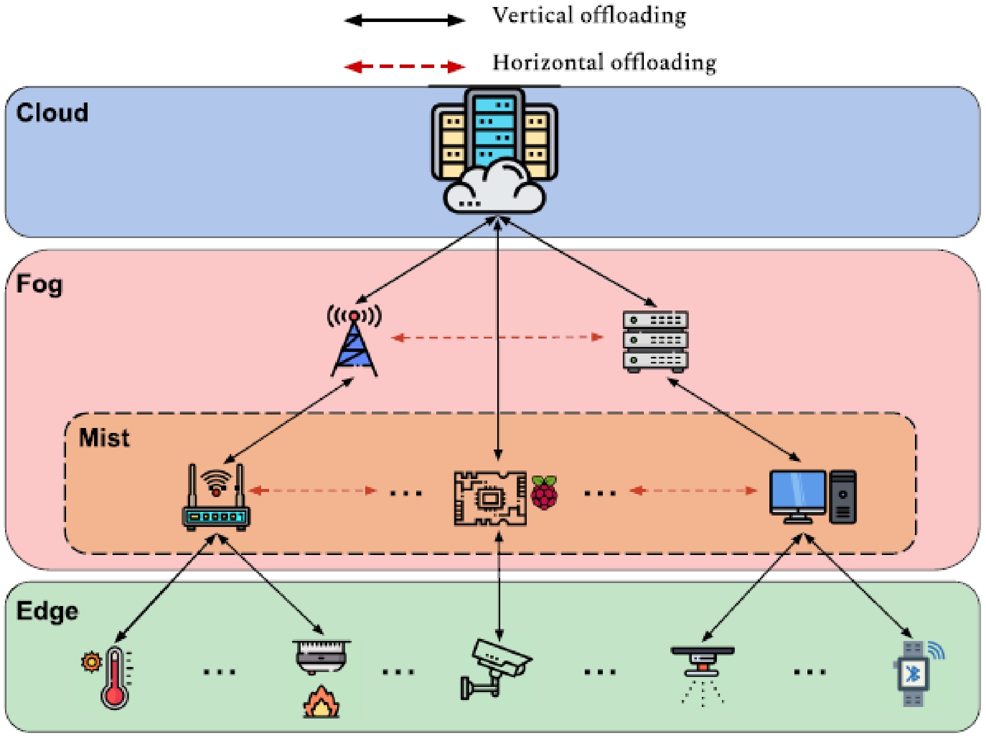


Fig. 8.1 Fog Computing Model (adapted from Iorga et al. (2017))

Example 5

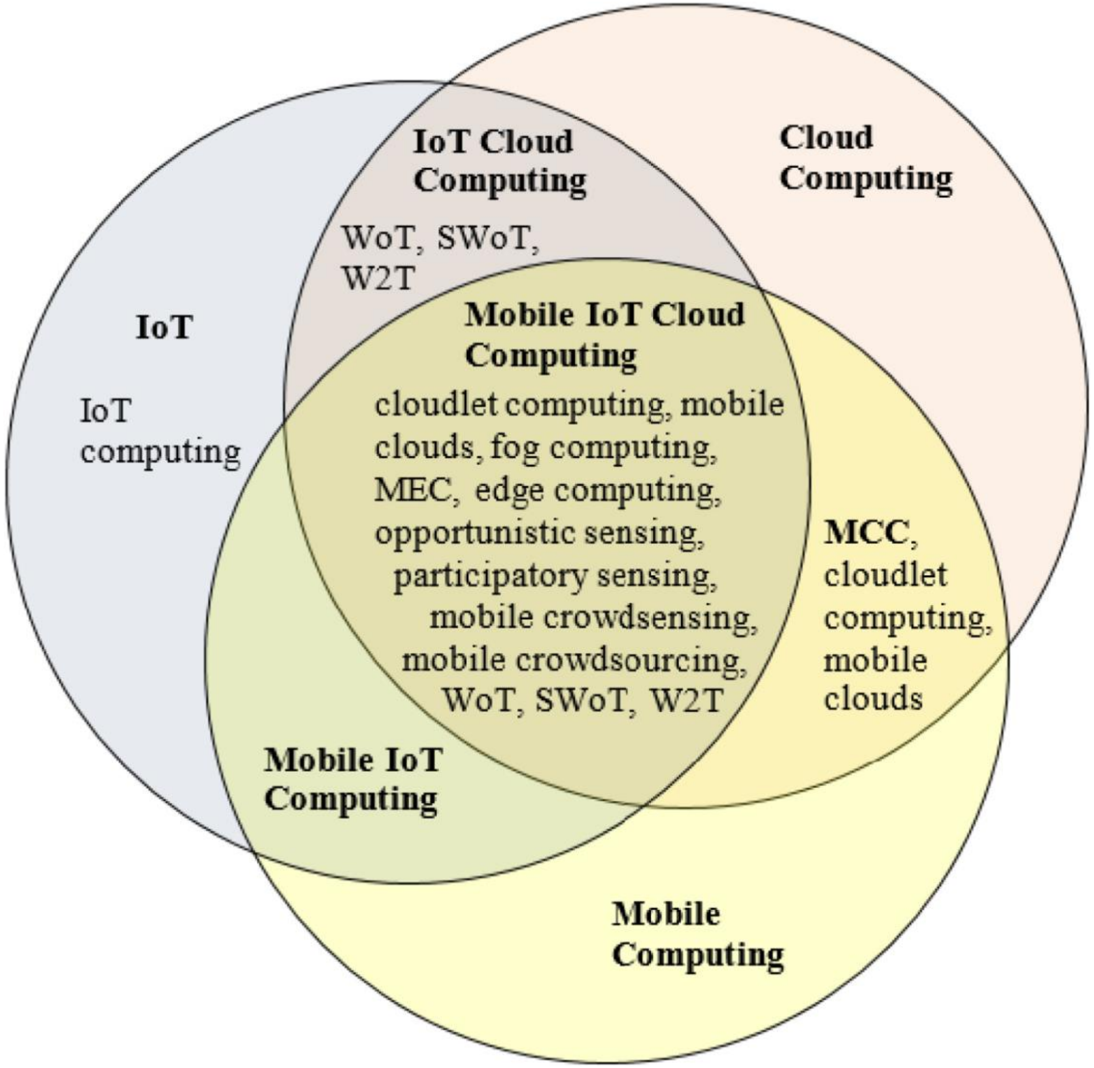

Cloud paradigm
Fog paradigm
IoT devices
Application: Traffic, VR game, Biometric
Task: Information storage
Application: Traffic
Task: Road status broadcast
Application: Traffic
Task: Road status update
Application: VR game
Task: Point calculation
Application: Traffic
Task: Road status update
Application: Biometric
Task: Identification
Application: Traffic
Task: Road status check
Application: VR game
Task: Account settings
Application: Parking monitor
Task: Empty lot detection
Application: Traffic
Task: Road status check
Application: VR game
Task: Account settings
Application: Traffic
Task: Traffic light management
VR box
Carputer
VR box
Finger print scanner
Road traffic unit
Carputer

Concept



- **Edge**: data processing is held immediately on the Edge IoT devices, reducing network latency and overheads.
- **Mist:** considers that the processing capabilities are close to data sources (e.g.access points, single-board computers). They can cooperate using clustering based on peer-to-peer communications.
- **Fog**: Compute resources are "relatively" close to data sources. Indeed, Fog nodes could be part of the access networks (e.g., servers, cloudlets, routers). Similarly to Mist nodes, Fog nodes can collaborate through clustering.
- **Cloud** provides huge computational/storage resources for the lower levels (Fog/Mist and Edge) when their capabilities are not sufficient..

IoT Computing Paradigms



Key Differences: Cloud Computing vs. Fog and Edge Computing

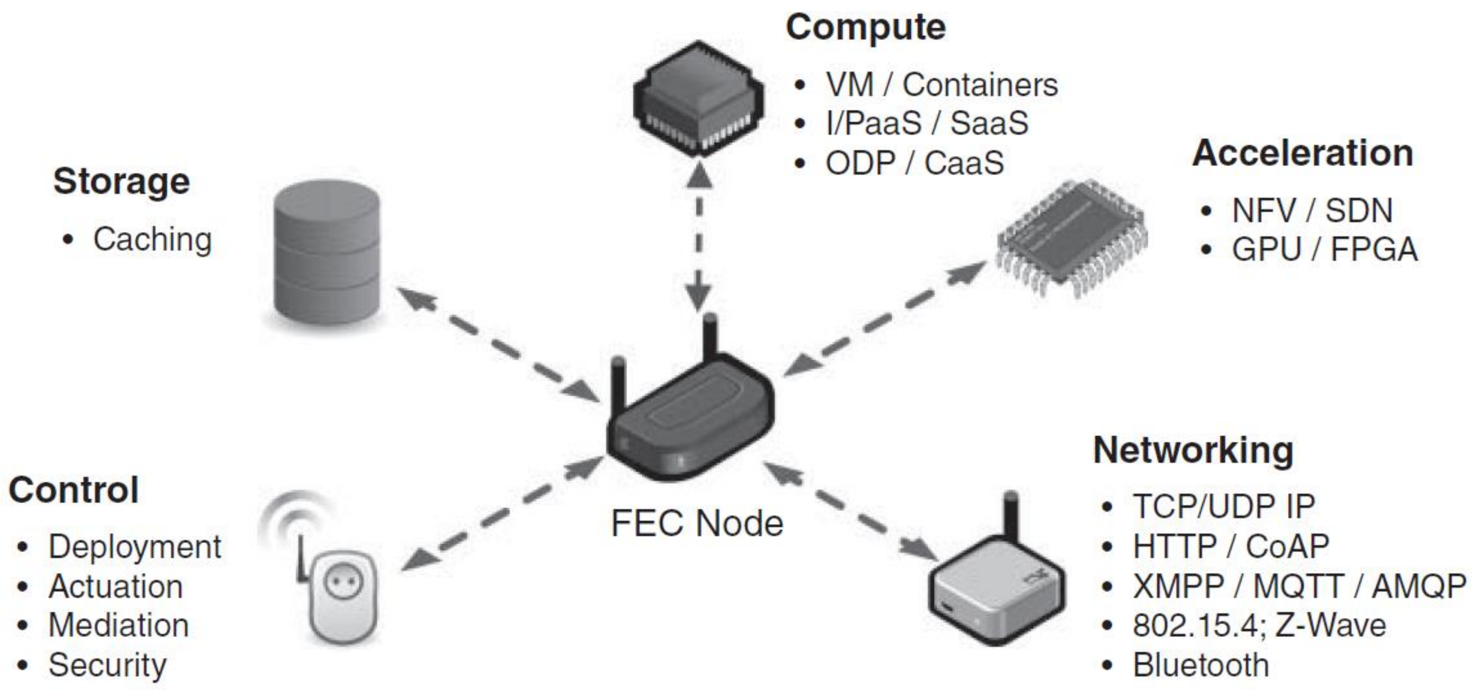
	Cloud Computing	Fog and Edge Computing
Architecture	Centralized (servers in data centers)	Distributed (via fog or edge nodes)
Location of server nodes	On the Internet (global)	At the edge of the network (local)
Location awareness	No	Yes
Latency	Medium	Low
Amount of nodes	Few (large servers)	Millions (of fog or edge nodes)
Support for mobility	Limited	Supported
Computing capabilities	Higher	Lower
Analysis	Long-term/deep analysis	Short-term
Connection	Internet	Various protocols and standards
Risk of failure	Medium (depending on the Internet)	Low (based on different connections)
Security	Medium	High (due to distributed architecture)

Advantages: SCALE

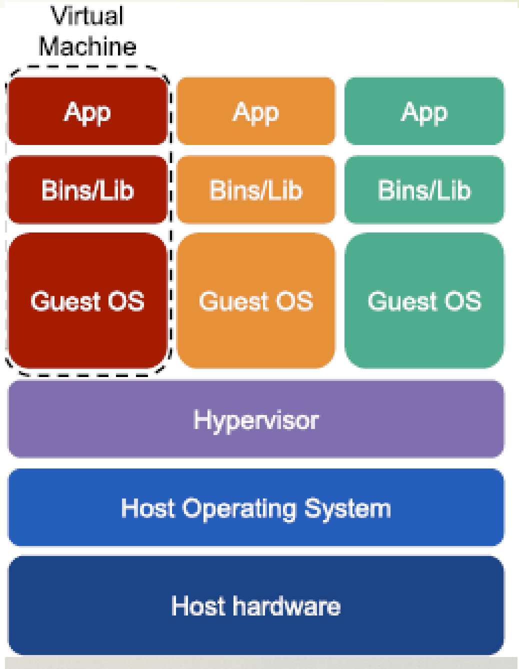
- **S**ecurity
- **C**ognition
- **A**gility
- **L**atency
- **E**fficiency

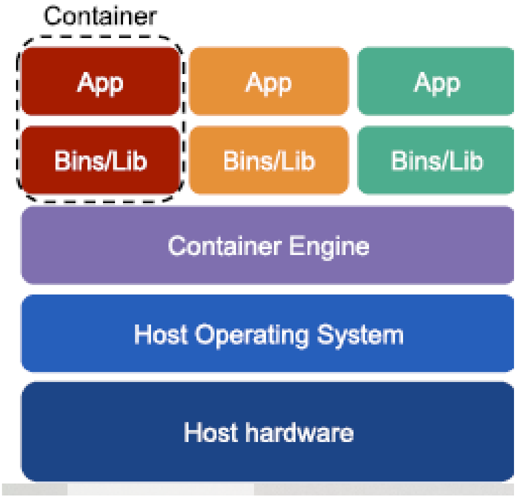
To achieve SCALE with SCANC

- Storage
- Compute:
  - 2 Models: I/PaaS and SaaS
  - 2 Approaches:
    - Hypervisor VMs
    - Container engines
  - SaaS:
    - on-demand processing
    - Context as a service (CaaS)
- Acceleration:
  - Networking
  - Computing
- Networking:
  - Vertical
  - Horizontal (wireless domain)
- Control:
  - Deployment
  - Actuation
  - Mediation (interaction)
  - Security



Virtualization vs Containerization





Virtualization paradigm (i.e., hardware virtualization) is characterized by higher complexity compared to the former as it makes use of a fully abstracted environment including the emulation of low-level (software-based) hardware management.

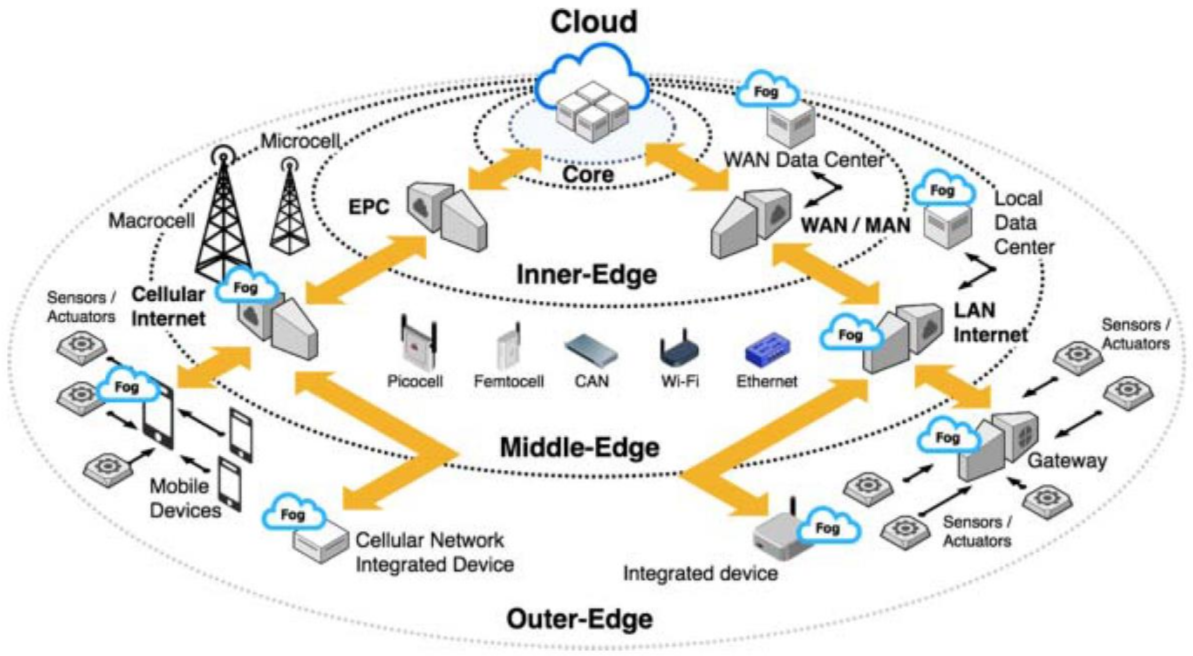
Containers or containerization paradigm, the abstraction of resources occurs at the host kernel whereas the hardware virtualization is below that level (i.e., the host kernel).

Learn more:
https://www.technicolor.com/sites/default/files/whitepapers/2018-edge-compute-and-software-life-cycle-management.pdf

Fog and Edge Computing (FEC)

Hierarchy of FEC

- Inner-Edge: the WAN-based cloud data center
- Middle-Edge
  - Local Area Network
  - Cellular Network
- Outer-Edge
  - Constraint Devices
  - Integrated Devices
  - IP Gateway Devices

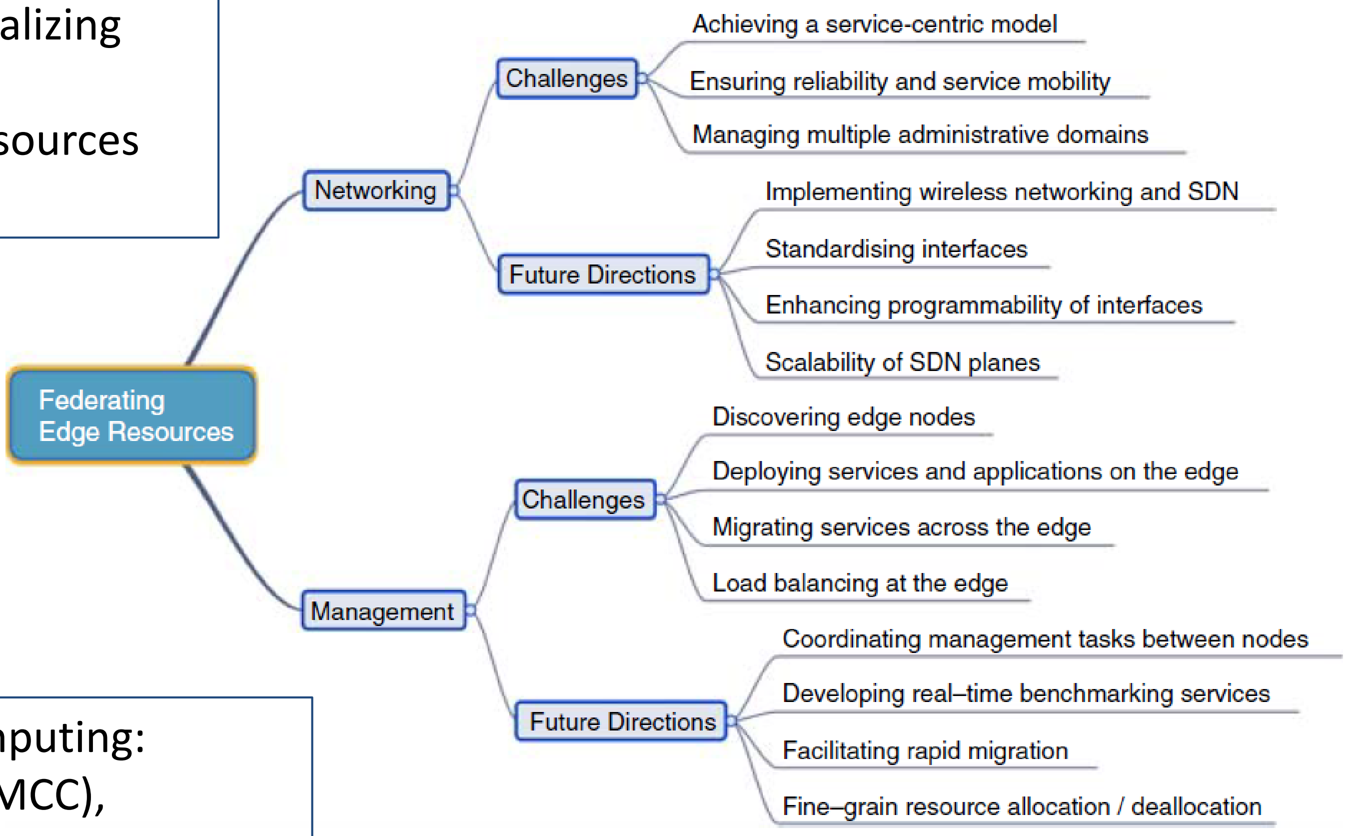


EPC: evolved packet core

Ref: Fog and Edge Computing: Principles and Paradigms

Networking and Management Challenges

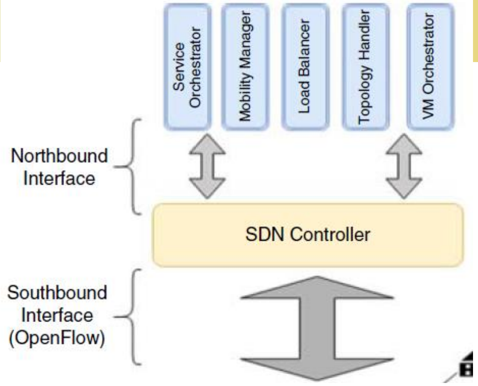
Edge to refer to a collection of technologies as decentralizing data center resources for bringing computational resources closer to the end user.

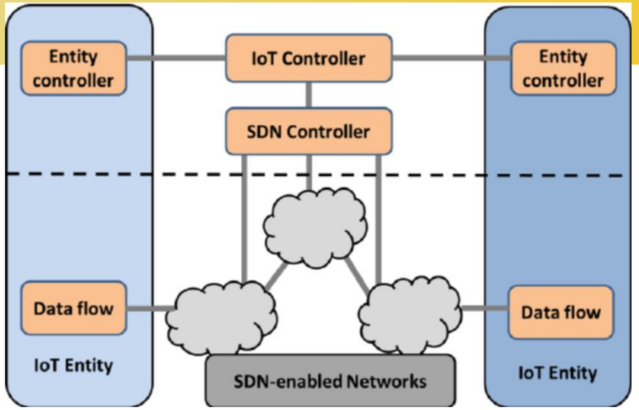


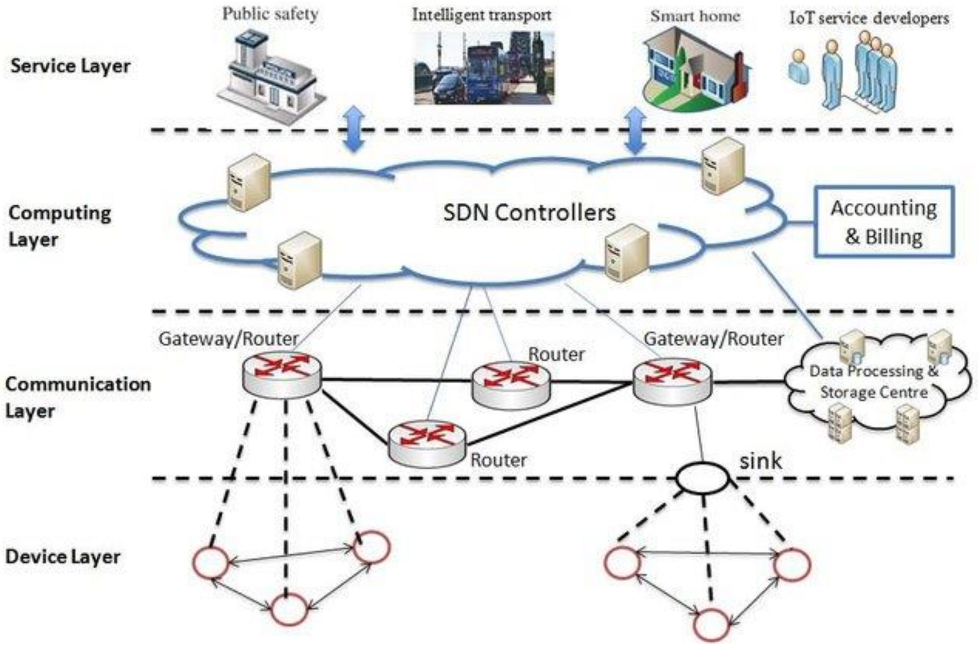
The Instances of Edge Computing:
Mobile cloud computing (MCC),
Cloudlets,
Fog computing,
Multi-access Edge Computing (MEC)

SDN: Software Defined Networking

SDN in IoT

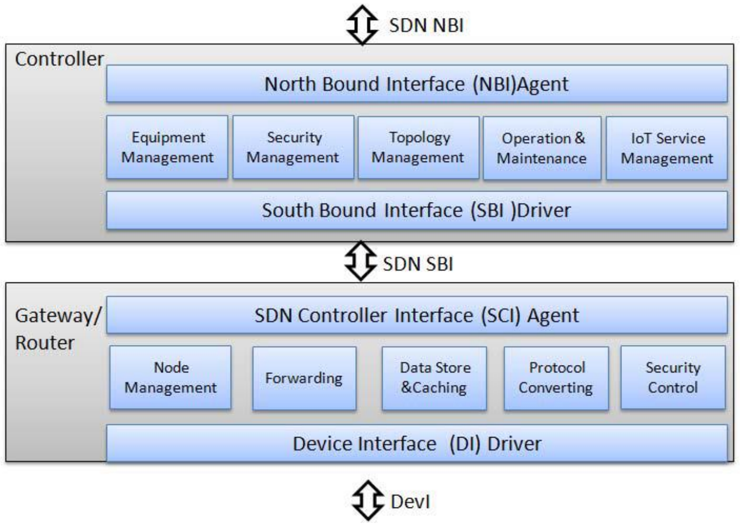






SDN: Software Defined Networking

SDN Controller Example



Networking Challenges

- We need to handle application layer handovers from one edge node to another.
- It's upon where the users are located and how request patterns are formed, the location of a service may change at any time under QoS satisfaction.

Networking challenge	Why does it occur?	What is required?
User mobility	Keeping track of different mobility patterns	Mechanisms for application layer handover
QoS in a dynamic environment	Latency-intolerant services, dynamic state of the network	Reactive behavior of the network
Achieving a service-centric model	Enormous number of services with replications	Network mechanisms focusing on "what" instead of "where"
Ensuring reliability and service mobility	Devices and nodes joining the network (or leaving)	Frequent topology update, monitoring the servers and services
Managing multiple administrative domains	Heterogeneity, separate internal operations and characteristics, different service providers	Logically centralized, physically distributed control plane, vendor independency, global synchronization

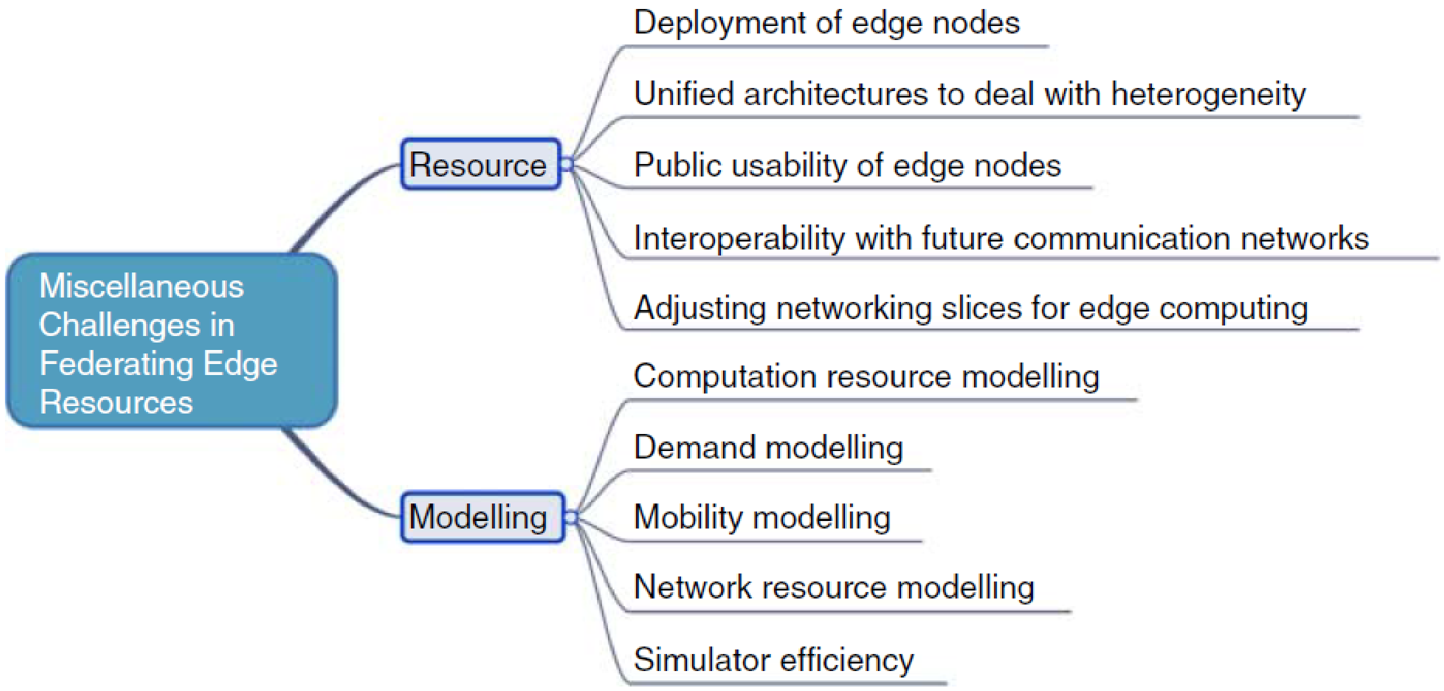
Edge Resources

Management Challenges

Management challenge	Why should it be addressed?	What is required?
Discovery of edge nodes	To select resources when they are geographically spread and loosely coupled	Lightweight protocols and handshaking
Deployment of service and applications	Provide isolation for multiple services and applications	Monitoring and benchmarking mechanisms in real-time
Migrating services	User mobility, workload balancing	Low overhead virtualization
Load balancing	To avoid heavy subscription on individual nodes	Auto-scaling mechanisms

Research Challenges

- In order to support multiple use cases and demands, various communication and resource models, and heterogeneous environments.



See more details ch3 [BS19] for research topics in:
energy consumption, performance, resource consumption, costs, QoS, Security

IoT Edge Examples

Microsoft Azure IoT Edge

https://azure.microsoft.com/en-us/services/iot-edge/

AWS Greengrass

https://aws.amazon.com/greengrass/

IBM Watson IoT

https://www.ibm.com/cloud/internet-of-things

Google

https://cloud.google.com/iot-core/

References

[VB17] Ovidiu Vermesan and Joel Bacquet, "**Cognitive Hyperconnected Digital Transformation: Internet of Things Intelligence Evolution**", River Publishers, 2017.

[BHC+18] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "**The Industrial Internet of Things (IIoT): An Analysis Framework**", Computers in Industry 101, October 2018.

[BS19] Rajkumar Buyya and Satish Narayana Srirama, "**Fog and Edge Computing: Principles and Paradigms**", Wiley Series on Parallel and Distributed Computing, 2019.

[LEA20] Perry Lea, "**IoT and Edge Computing for Architects: Implementing edge and IoT systems from sensors to clouds with communication systems, analytics, and security"**, 2nd Edition, Packt Publishing, 2020.

Simplilearn
https://www.youtube.com/watch?v=KeaeuUcw02Q
Google
https://www.youtube.com/watch?v=-T9MNR-Bl8I
Azure IoT Edge:
https://www.youtube.com/watch?v=01kHKNN3z-0
https://www.youtube.com/watch?v=rjfCNakISBM (Implement)
Eseye
https://www.youtube.com/watch?v=knQbAEhipxM (IoT Ecosystem)
Atos
https://www.youtube.com/watch?v=cxFJWxQRRDY (IoT Ecosystem Security

Oracle:
https://www.youtube.com/watch?v=HoGV35vpTEY
NGIOT:
https://horizon-europe-cloud-edge-iot.b2match.io/
https://www.ngiot.eu/from-cloud-to-edge-to-iot-for-european-data/
https://www.youtube.com/playlist?list=PLBrivHE6_rsdWf9F9KN-MyFZ9b57RjcDm

More References: IoT + ML

- Creating a Machine Learning IoT application on Raspberry Pi with Node-RED and TesnsorFlow.js
  https://www.youtube.com/watch?v=6sFrQaDtK5Q
- IoT Edge examples
  https://adv-corp-test.azurewebsites.net/resources/?solution=iot-edge-intelligence-software-solutions
- TinyML study group
  https://github.com/scaledown-team/study-group
- IBM
  https://developer.ibm.com/technologies/iot/tutorials/iot-cognitive-iot-app-machine-learning/
- From Cloud to Edge to IoT
  https://www.ngiot.eu/wp-content/uploads/sites/73/2021/03/NGIOT-Workshop_25Feb2021-RRiemenschneider.pdf
- AI + Sensor
  https://www.electropages.com/blog/2021/02/how-ai-taking-sensors
- Edge Intelligence
  https://rvce.edu.in/sites/default/files/RVJSTEAM-ISSUE1.pdf
- Computing at the Edge of IoT – Google Developers – Medium
  https://www.epanorama.net/blog/2018/02/18/computing-at-the-edge-of-iot-google-developers-medium/

ITCS447
Lecture 10

IoT Dashboards
Cloud (eg. NETPIE, AWS, IBM Cloud)
Opensource (eg. Thingsboard, Node-RED)
Smartphone (eg.Blynk)

LINE Notification

Asst. Prof. Dr. Thitinan Tantidham

มหาวิทยาลัยมหิดล
Mahidol University

Outline

- Components in IoT Platform
- IoT for Monitoring and Control
- IoT for Proactive Protection
- IoT Data Visualization (IoT Dashboards)
- IoT Dashboards on Edge – NODE-RED
- IoT Dashboard on Cloud – NETPIE
- IoT Dashboard on Smartphone – Blynk
- Data Notification - LINE Notify

Components in IoT Platform

Database
Repository that stores the important data sets

External interfaces
APIs, SDKs and gateways that act as interfaces for 3rd party systems (e.g., ERP, CRM)

Analytics
Algorithms for advanced calculations and machine learning

Additional tools
Further development tools (e.g., app prototyping, access management, reporting)

Data visualization
Graphical depiction of (real-time) sensor data

(Dashboards)

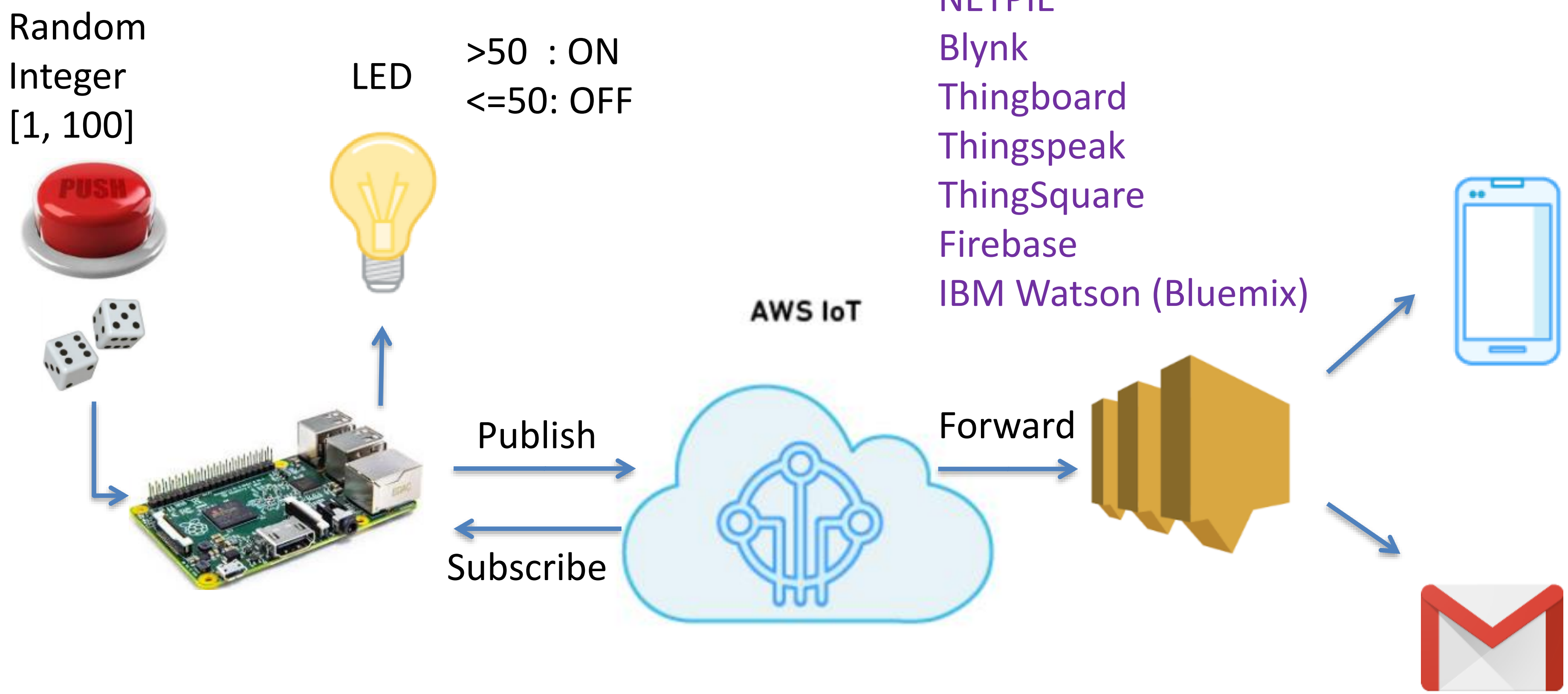
Processing & action management
Rule engine that allows for (real-time) actions based on incoming sensor & device data

Device management
Backend tool for the management of device status, remote software deployment and updates

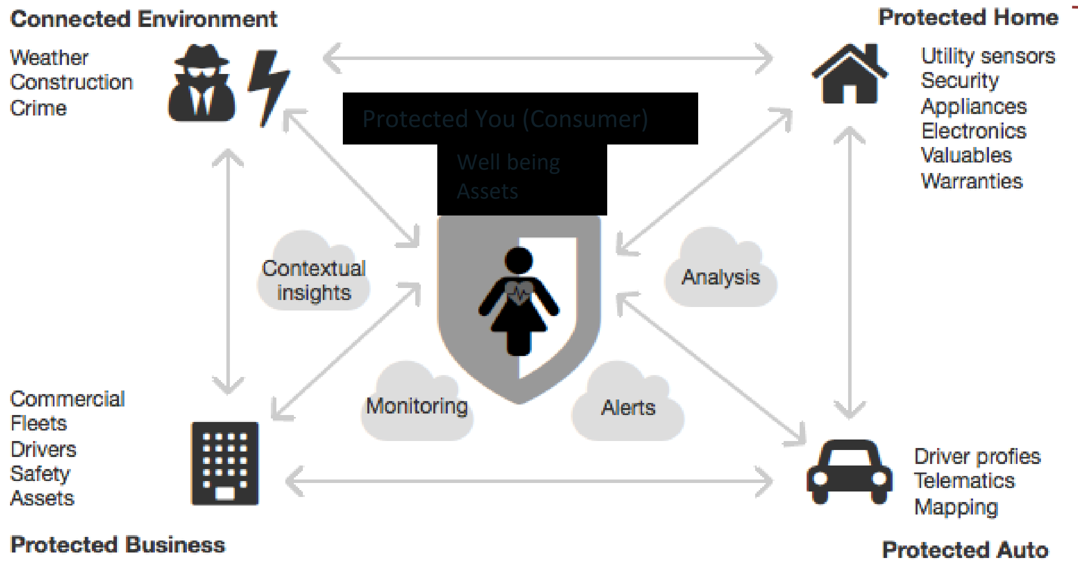
Connectivity & Normalization
Agents and libraries that ensure constant object connectivity and harmonized data formats

IoT for Monitoring and Control

AWS (Amazon Web Service)
NETPIE
Blynk
Thingboard
Thingspeak
ThingSquare
Firebase
IBM Watson (Bluemix)



IoT for Proactive Protection



<u>Water Leak Detection</u>

- Reduced claims
- Real time notifications
- Personalized risk assessment
- Satisfied customers

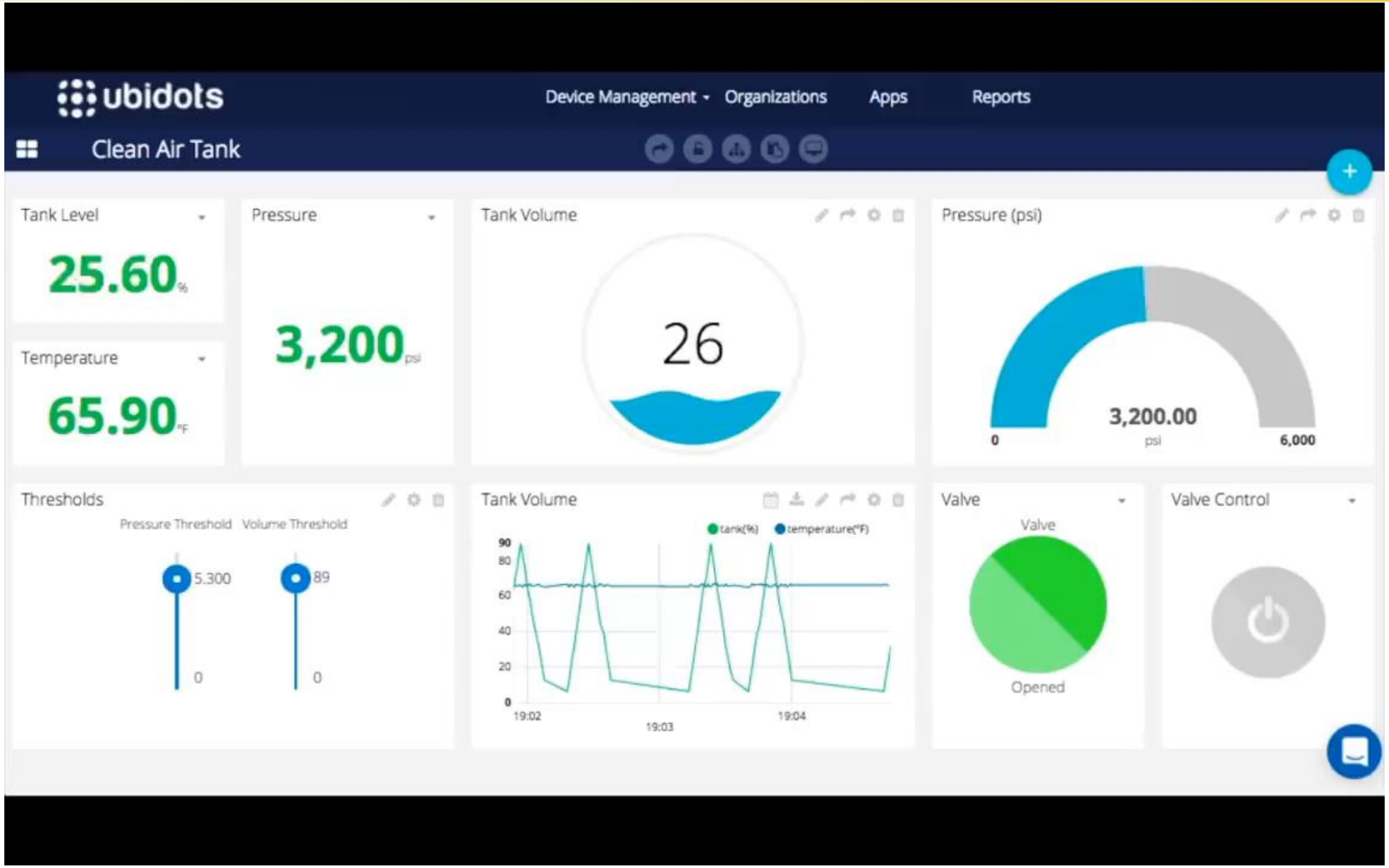
IoT Data Visualization (IoT Dashboards)

- For the IoT, or any control system, IoT dashboard is the key HMI (Human-Machine Interface) component that organizes and presents digital information from our physical world into a simply understood display on a computer or mobile device.

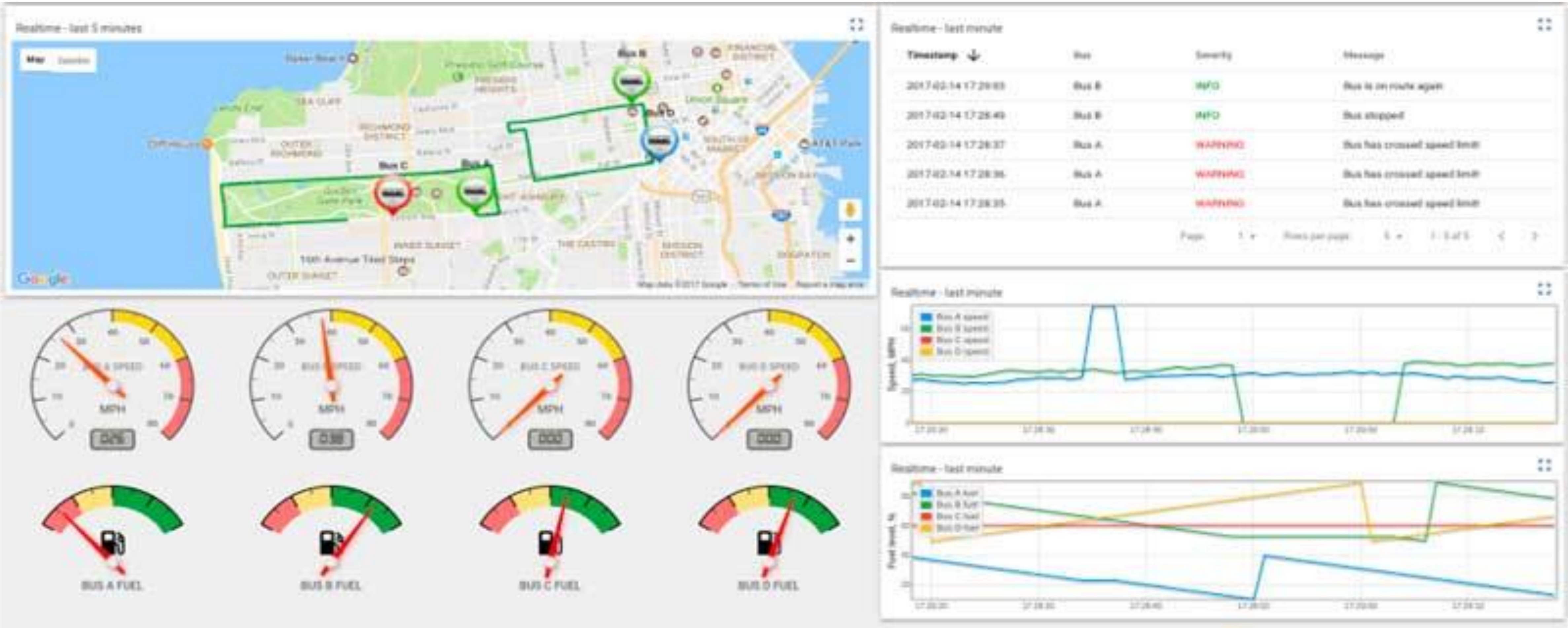
- With the help of IoT Dashboards, users and operators can (remotely) monitor and control specific assets and processes, and depending on safety requirements, access and control an environment from anywhere in the world.

- IoT dashboards populated with <u>graphs</u>, <u>charts</u>, <u>control switches</u>, <u>maps</u>, <u>tables</u>, and countless other widgets are the digitals tools we use to visualize and display data coming from the physical world to our computers.

Examples: ubidots (Monitoring and Control)

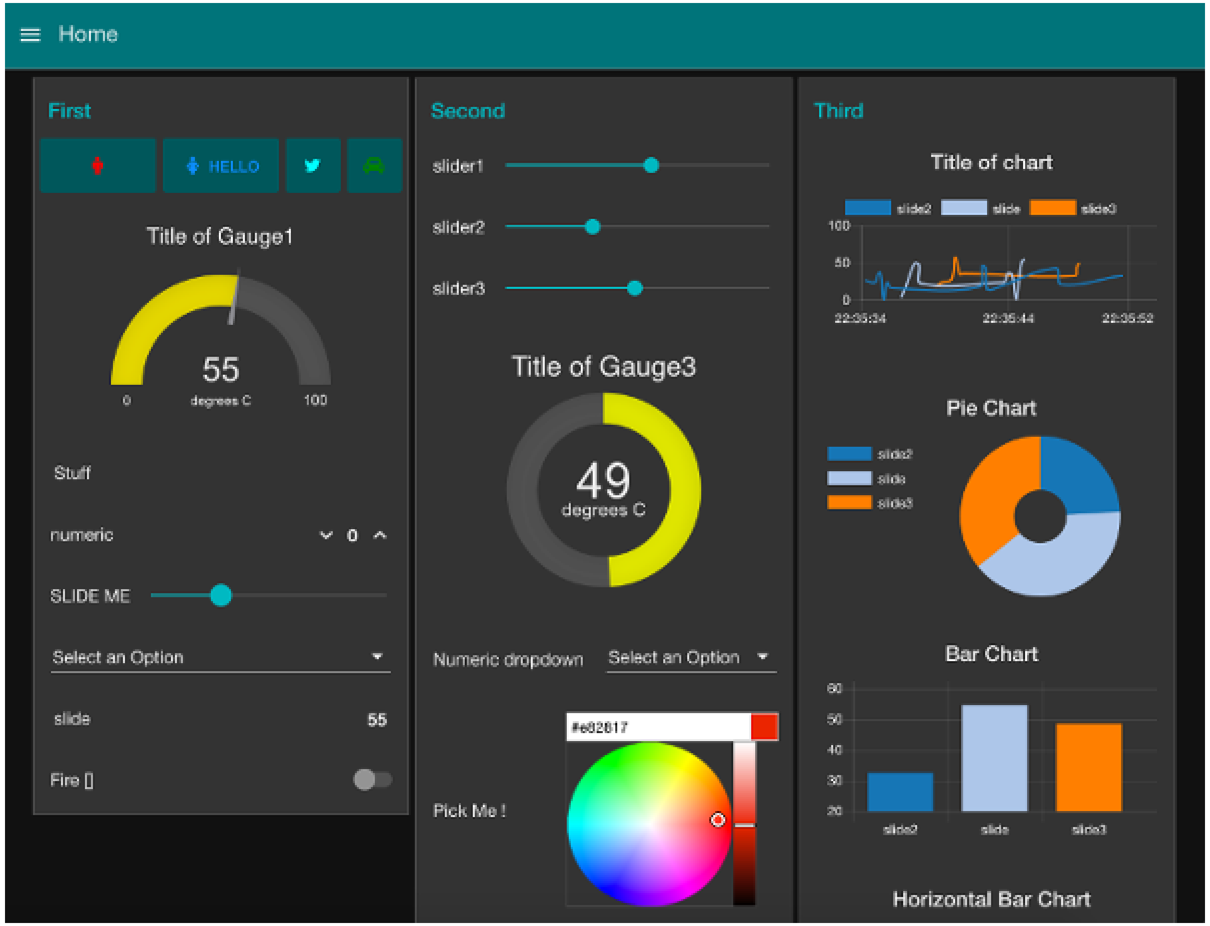


Examples: with Map (Regions) e.g. Thingspeak



Node-RED

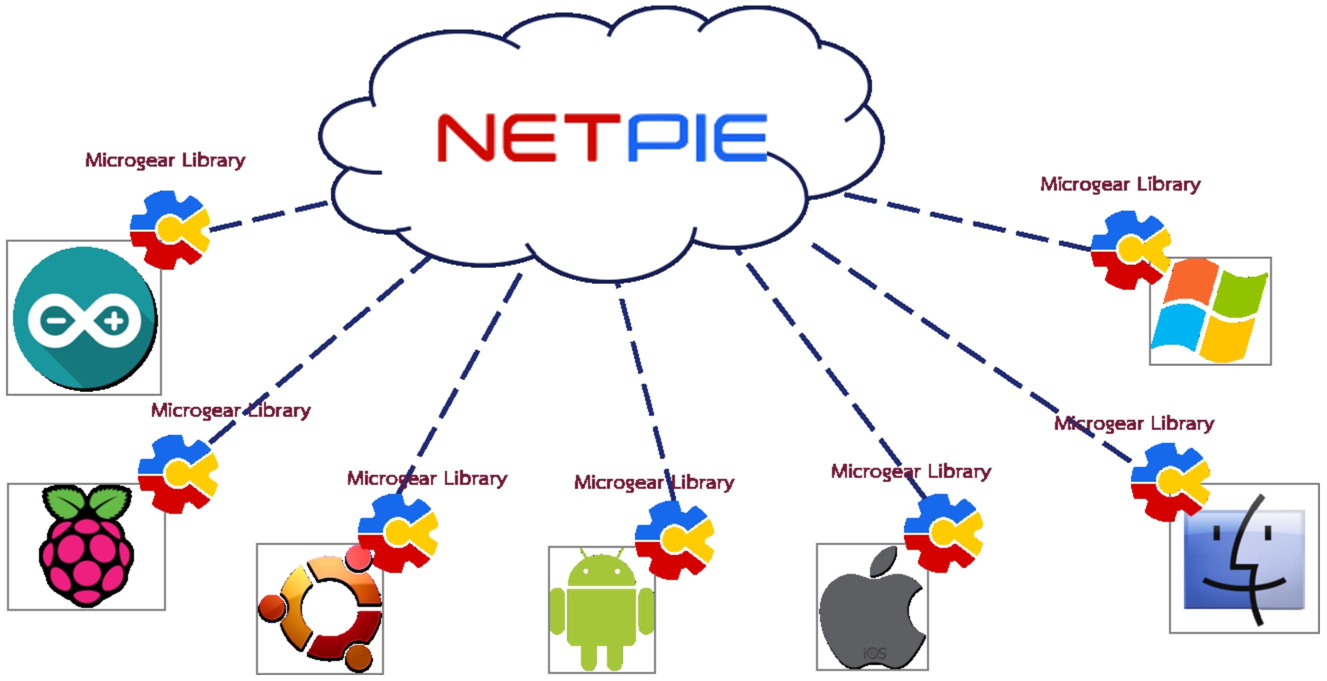
https://nodered.org/



NETPIE

http://netpie.io

Ref: NETPIE2015-WS_v25_TH.pdf or NETPIE2015_WS_vx_EN.pdf



NETPIE2020 vs NETPIE2015

	NETPIE 2020	NETPIE 2015
Design Philosophy	Platform - Centric	Device – Centric
Commercial Ability	Commercial – Ready	Need to re-program Hardware migrate to commercial platform
Suitable Usage	Consumer product, Project-based (3rd-Party)	Project-based
Target User	IoT consumer product makers, Hobbyists, Students	Makers, hobbyists, students
Communication Protocol	MQTT, HTTP	Microgear
Programming Language	Any languages with MQTT library support	Limited to Microgear library
Hardware Support	Unlimited as long as it supports MQTT	Limited to those with Microgear support
Device Identity and Group	No APPID Device identity and group can be adjusted after product is sold/installed	Use APPID Device identity and group must be programmed into firmware
Rate Limit	Allow burst	Everyone is subject to the same rate limit
Trigger	Can set trigger action in cloud platform	Set trigger action inside IoT devices

NETPIE on ESP32 (1)

NETPIE2015

- NETPIE2015 library

```
#include "MicroGear.h"
```

- Application related keys

```
#define APPID   ”..."  // from NetPie AppID
#define KEY     ”...” // Application Key from NetPie
#define SECRET  ”...” // Session Key from NetPie
#define ALIAS   "PM25” // Alias of this device
```

NETPIE on ESP32 (2)

NETPIE2015

- Feed related keys

```
#define FEEDID "..." // Feed ID from NetPie
#define FEED_INTERVAL 2000 // How often you are sending data to feed
#define PRES_NAME "pres" // Same feed name on NetPie for pressure
#define TEMP_NAME "temp" // Same feed name on NetPie for temperature
#define HUMID_NAME "hum” // Same feed name on NetPie for humidity
#define PM10_NAME "pm10" // Same feed name on NetPie for PM 10
#define PM25_NAME "pm2_5" // Same feed name on NetPie for PM 2.5
#define PM1_NAME "pm1_0" // Same feed name on NetPie for PM 1.0
```

- Application related keys

```
#define APPID "..."  // from NetPie AppID
#define KEY "..." // Application Key from NetPie
#define SECRET  "..." // Session Key from NetPie
#define ALIAS  "PM25" // Alias of this device
```

NETPIE on ESP32 (3)

NETPIE2015

- Instantiate a Microgear object

```
MicroGear microgear(client); // client is WiFiClient object
```

- Configure microgear object with keys and connect to NETPIE2015

```
microgear.on(CONNECTED, onConnected); // Call onConnected() when NETPIE connection is established
microgear.init(KEY, SECRET, ALIAS); // Initiate Microgear with KEY, SECRET and also set the ALIAS here
microgear.connect(APPID); // connect to NETPIE with a specific APPID
```

- Method connected() can be used to check if the device currently connects to NETPIE or not

```
microgear.connected()
```

NETPIE on ESP32 (4)

NETPIE2015

- Create a string to be transmitted to NETPIE 2015's feed

```
sprintf(dataBuffer,"%s:%d,%s:%d,%s:%d,%s:%.2f,%s:%.2f,%s:%.2f",
    PM1_NAME, data.PM_AE_UG_1_0, PM25_NAME, data.PM_AE_UG_2_5, PM10_NAME,
    data.PM_AE_UG_10_0, PRES_NAME,pres,
    TEMP_NAME, temp, HUMID_NAME, hum);
```

- Write the created data string to feed

```
microgear.writeFeed(FEEDID, dataBuffer);
```

- Write each data value to NETPIE 2015's application

```
microgear.chat(PM1_NAME, data.PM_AE_UG_1_0); // PM 1.0
microgear.chat(PM25_NAME, data.PM_AE_UG_2_5); // PM 2.5
microgear.chat(PM10_NAME, data.PM_AE_UG_10_0); // PM 10
microgear.chat(PRES_NAME, pres); // Pressure
microgear.chat(TEMP_NAME, temp); // Temperature
microgear.chat(HUMID_NAME, hum); // Humidity
```

NETPIE on ESP32 (5)

NETPIE2015

- **onConnect()** event handler function

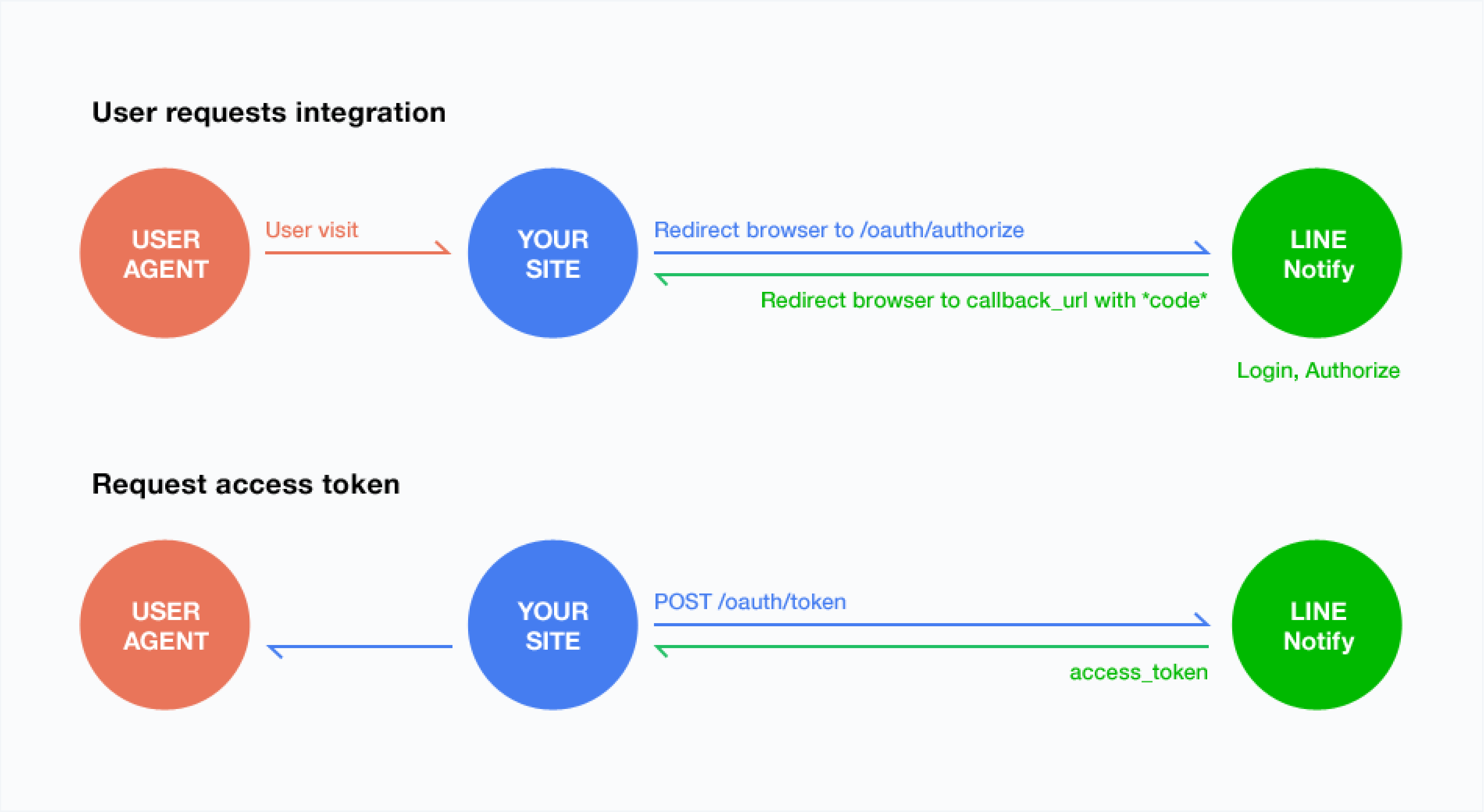
```
void onConnected(char *attribute, uint8_t* msg, unsigned int msglen)
{
}

/* Hook onConnected() to the "on connect" event
 (Automatically call onConnected() when the device connects to NETPIE) */
microgear.on(CONNECTED, onConnected);
```

- Make NETPIE know the alias of this device

```
microgear.setAlias(ALIAS); // Set the alias of this microgear ALIAS
```

LINE Notify


User requests integration
USER AGENT
User visit
YOUR SITE
Redirect browser to /oauth/authorize
LINE Notify
Redirect browser to callback_url with *code*
Login, Authorize
Request access token
USER AGENT
YOUR SITE
POST /oauth/token
LINE Notify
access_token

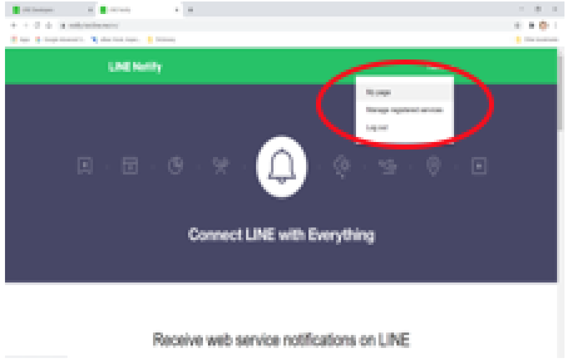
LINE Notify

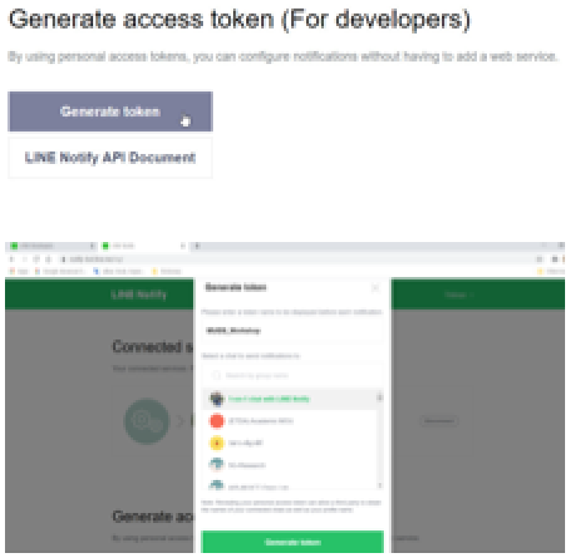
- We can use one LINE account to notify and send a message to the other LINE accounts.
- We can Push Notification from a device or an application to a Line Account or Line Group
- First, we need to Log in at https://notify-bot.line.me/en/ via our own Line Account
  - Then go to “My page” and to “Generate token”

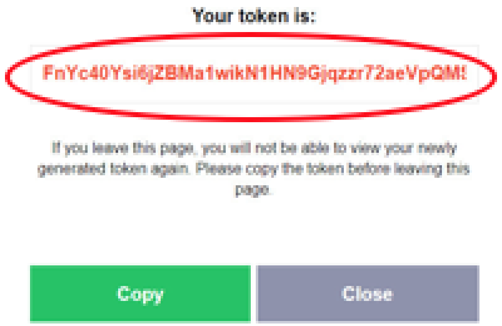
  Note: This Token is linked to this Line Account or this Line Group which we will send a message to

  - Copy the Token for your program

Token Generation







Line Notify

From a Device

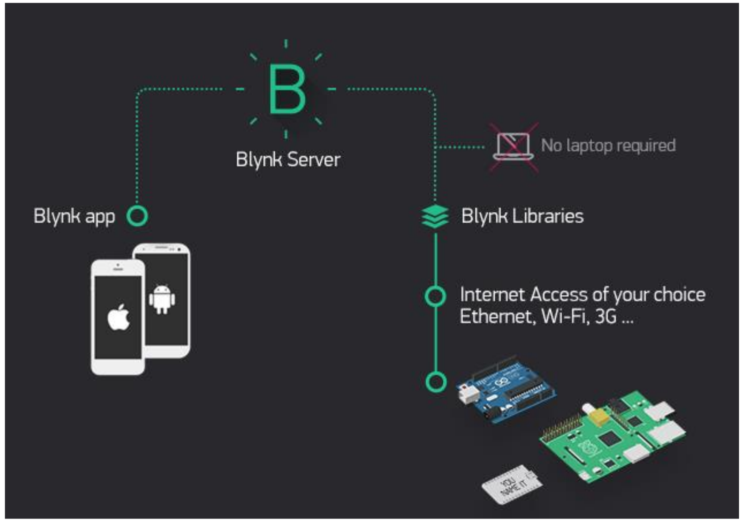
- Send a message to notify-api.line.me/api/notify with Token

```
POST /api/notify HTTP/1.1
Host: notify-api.line.me
Authorization: Bearer  <Line Token>
Cache-Control: no-cache
User-Agent: ESP32
Content-Type: application/x-www-form-
urlencoded
Content-Length: <Lenght of Message>

message=<Message>
```

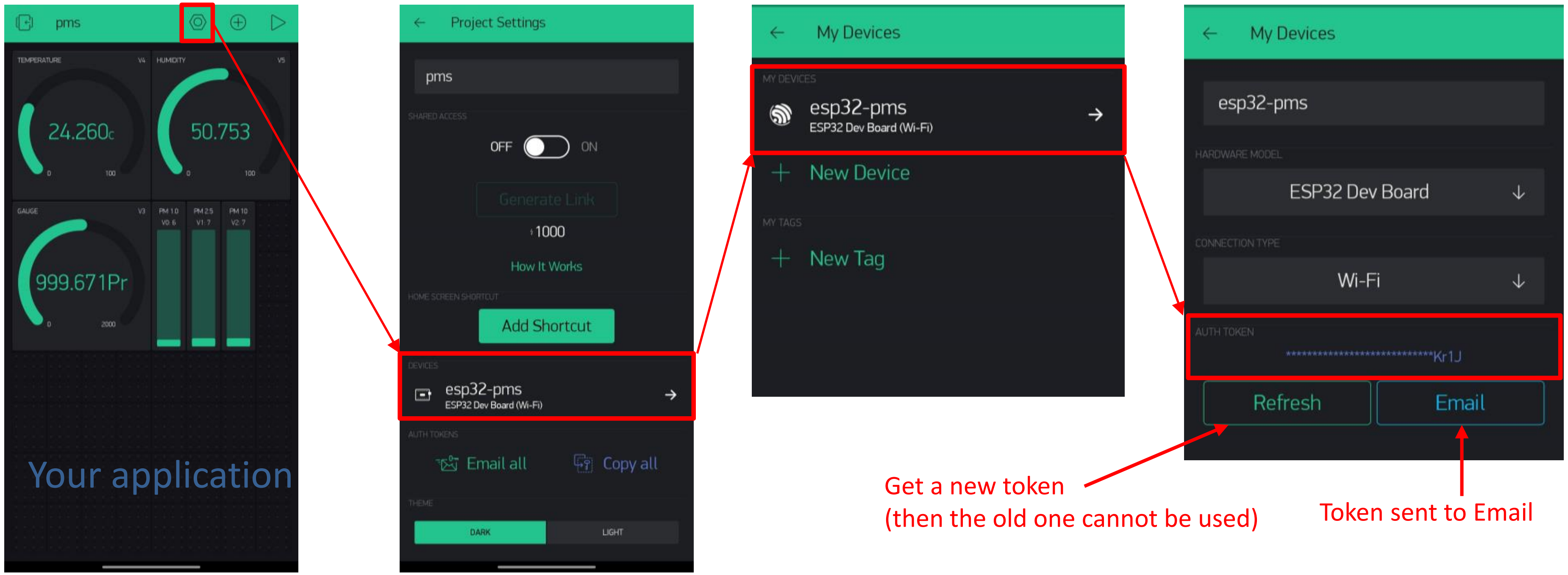
https://blynk.io

- Blynk is an APP for Android or iPhone (or tablets).
- It is a configurable control panel to read and control various GPIO devices on your hardware. It works across the internet and through firewalls. because it uses a Blynk Server out on the cloud.

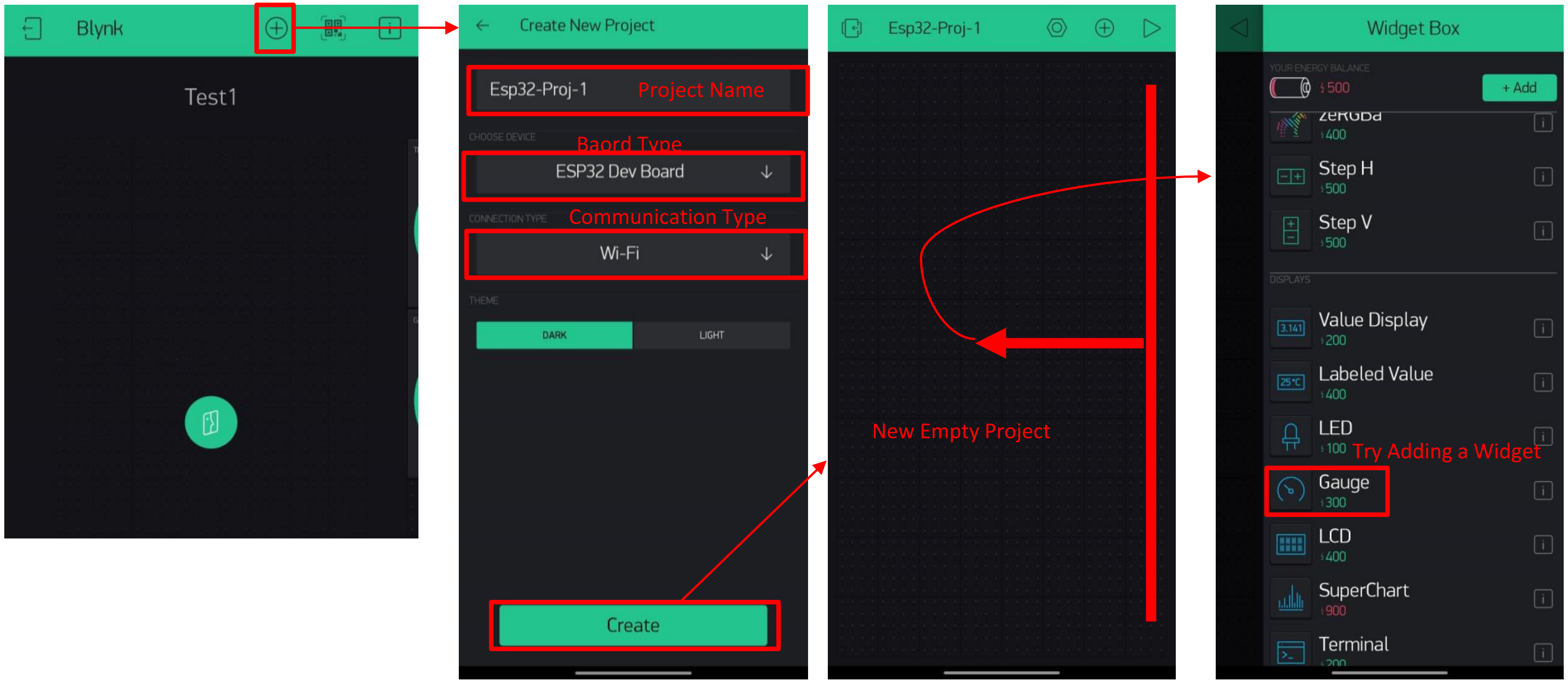


Blynk on ESP 32 (1)

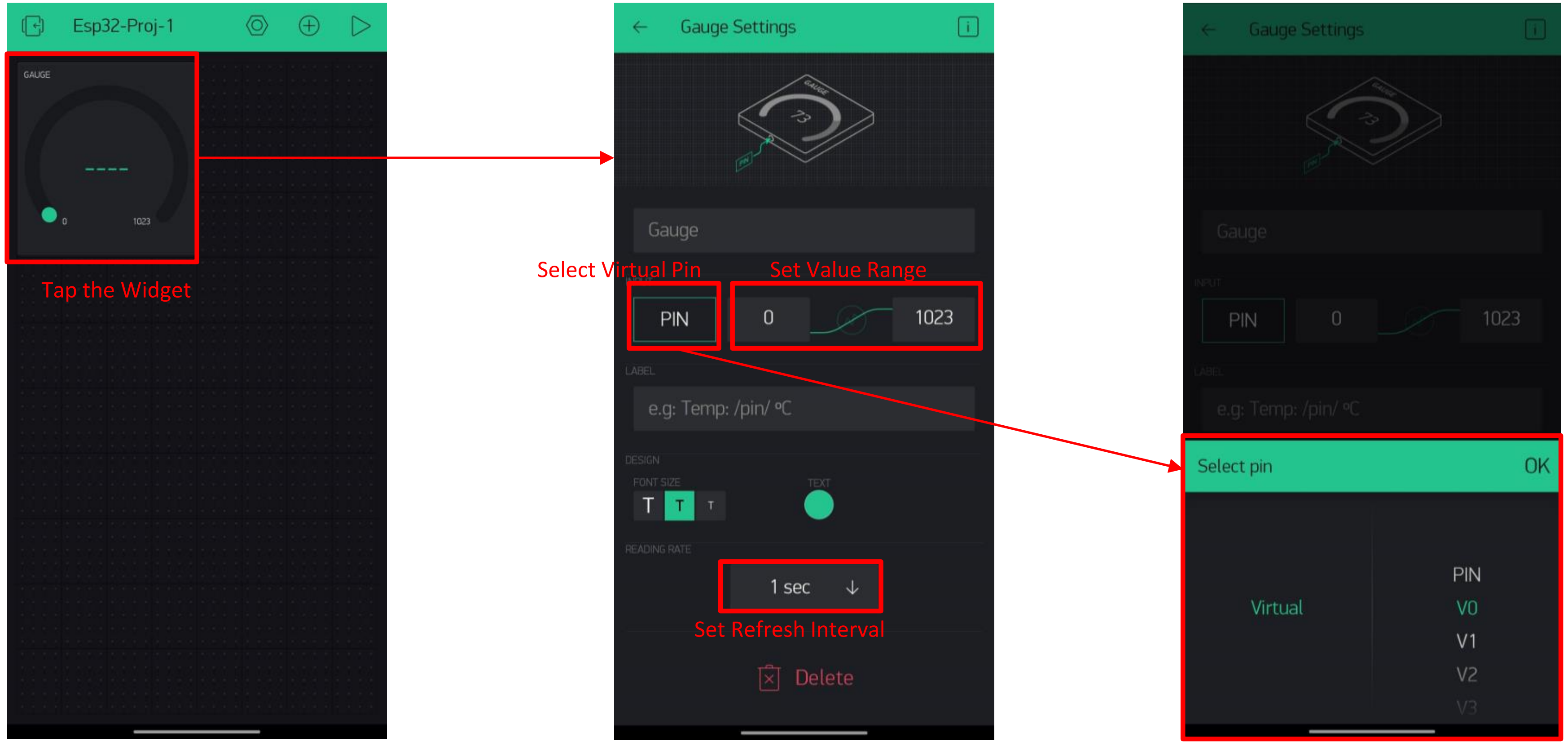
- You can connect your Blynk application to your ESP 32 device by using **Auth Token** which can be found by following the steps below.



Creating Blynk Project (1)

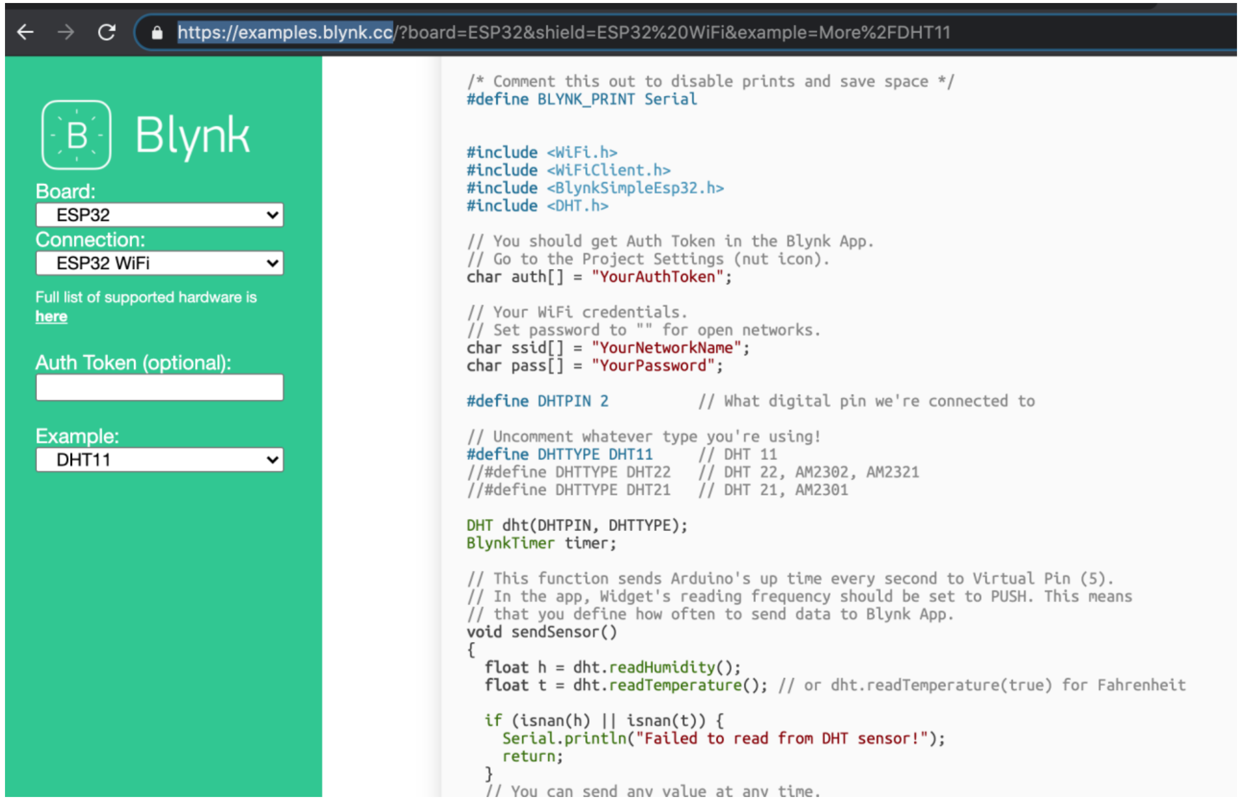


Creating Blynk Project (2)



Blynk on ESP 32 (2)

- Example code on https://examples.blynk.cc/

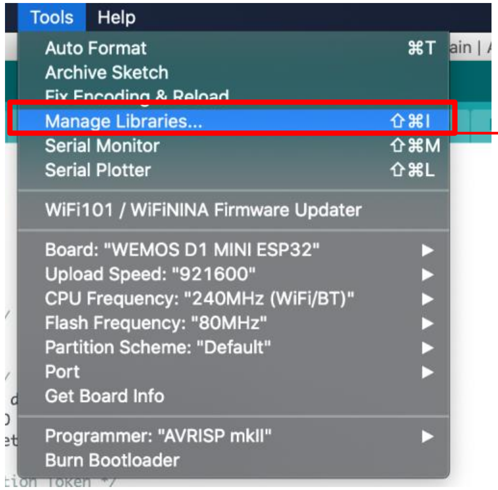


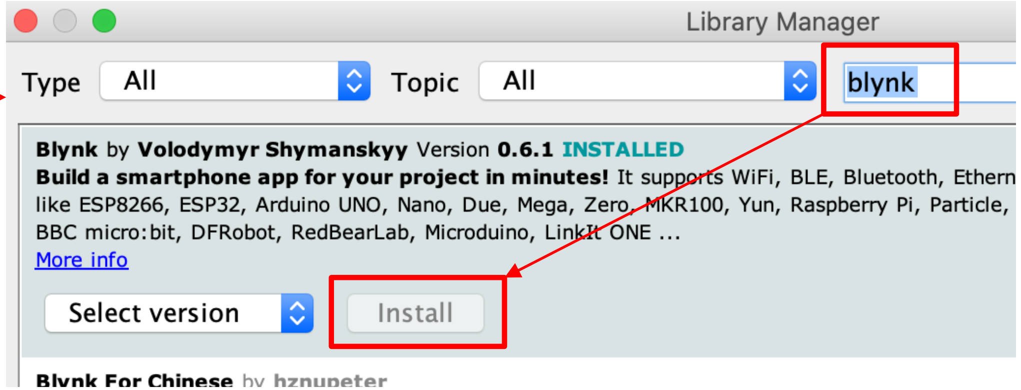
Blynk on ESP 32 (3)

- Blynk library

```
#include <BlynkSimpleEsp32.h>
```

- Installing the library to Arduino IDE





Blynk on ESP 32 (4)

- Configure the Blynk token and connect to Blynk server

```
bool blynkConnected = false; // Flag determining if this device can connect to Blynk
char auth[] = "..."; //Your Authentication Token hear
Blynk.config(auth);
blynkConnected = Blynk.connect(); // connect to Blynk
```

- Write data to each widget in your Blynk application

```
Blynk.run();
Blynk.virtualWrite(V0, data.PM_AE_UG_1_0); // Write PM1.0 value to virtual pin V0
Blynk.virtualWrite(V1, data.PM_AE_UG_2_5); // Write PM2.5 value to virtual pin V1
Blynk.virtualWrite(V2, data.PM_AE_UG_10_0); // Write PM10 value to virtual pin V2
Blynk.virtualWrite(V3, pres); // Write pressure value to virtual pin V3
Blynk.virtualWrite(V4, temp); // Write temperature value to virtual pin V4
Blynk.virtualWrite(V5, hum); // Write humidity value to virtual pin V5
```

Blynk on ESP 32 (5)

- This configuration option is used when you connect to WiFi seperately using WiFi library.

```
bool blynkConnected = false; // Flag determining if this device can connect to Blynk
char auth[] = ”..."; //Your Authentication Token hear
Blynk.config(auth);
blynkConnected = Blynk.connect(); // connect to Blynk
```

- Alternatively, Blynk can connect to WiFi and Blynk in one command in case you do not use a separate WiFi library for WiFi connection

```
char ssid[] = ”..."; // Your WiFi SSID.
char pass[] = ”..."; // Set password to "" for open networks.
char auth[] = ”..."; //Your Authentication Token hear
Blynk.begin(auth, ssid, pass);
```

References and Tutorials

- https://nodered.org/docs/tutorials/
- https://github.com/thingsboard/thingsboard

- https://docs.aws.amazon.com/iot/latest/developerguide/iot-dg.pdf
- https://www.ibm.com/cloud/internet-of-things
- https://thingspeak.com/

- https://2015.netpie.io/tutorials#gsc.tab=0
- https://netpie.io/tutorials

- https://notify-bot.line.me/doc/en/

- https://examples.blynk.cc/

ITCS447
Lecture 10
Introduction to NODE-RED

Asst. Prof. Dr. Thitinan Tantidham

มหาวิทยาลัยมหิดล
Mahidol University

Introduction to Node-RED

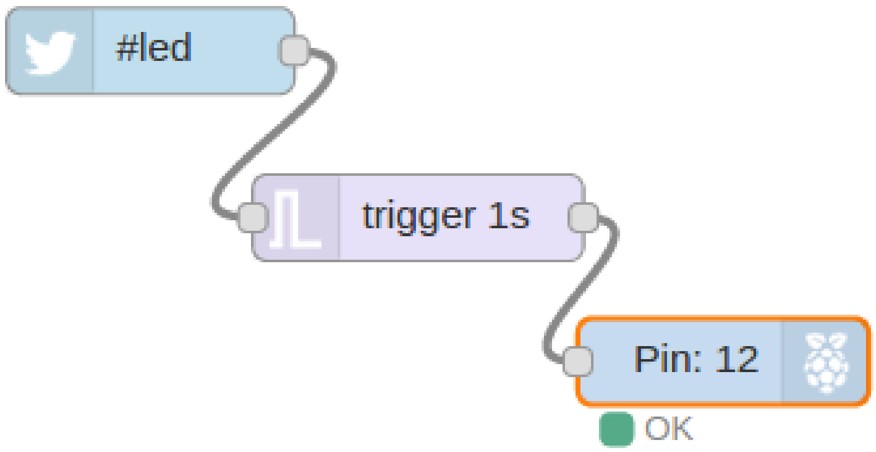
- Originally developed as an open source project at IBM in late 2013, to meet their need to quickly connect hardware and devices to web services and other software – as a sort of glue for the IoT – it has quickly evolved to be a general purpose an IoT visual programming tool.
- It allows developers to connect predefined code blocks, known as ‘nodes’, together to perform a task and to reuse Node-RED code for a wide variety of tasks.

- The connected nodes, usually a combination of input nodes, processing nodes and output nodes, when wired together, make up a ‘flows’.

- The flow can be represented in json.
- Many node can be functioned in Javascript.
- Node-RED as “a browser based on flow language” runs on pi or a server.

Node-RED Functionalities on Raspberry Pi (RPi)

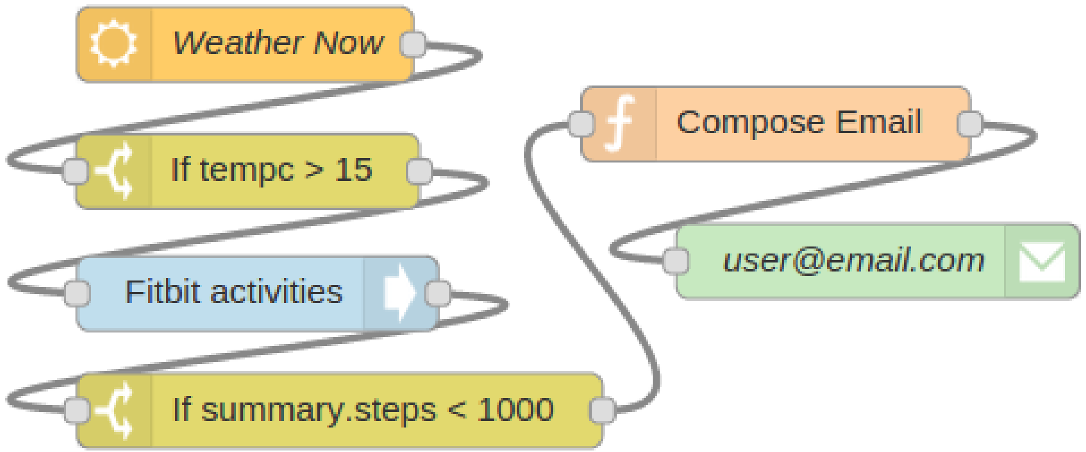
- Access your RPi GPIOs;



- The Twitter node is a built-in node in Node-RED and hides all of the complexity of using the Twitter API. It can be configured with a user's account credentials and a variety of search strings, in our case it's simply looking for hashtag '#led'.
- When the Twitter node sees the tag either in a tweet to the user, or the public tweet feed, it creates a new message with the details of the tweet, which is forwards to the next node in the flow.
- The trigger node is to wait for any message on its input. When it receives a message, it 'triggers', and sends a message on its output with the value "1" as the message body. It then waits 1 second and sends a second message with the value "0" in the message body.
- The trigger node controls the input/output on the RPi every 1 s.
- If RPi Pin12 is wired up with an LED connected to Pin 12, the gpionode going to on and off according to "1" and "0"

https://raw.githubusercontent.com/SenseTecnic/nrbookflows/master/lesson1/1-1_twitter.json

Node-RED Functionalities



- Openweather
- Switch
- Fitbit

https://raw.githubusercontent.com/SenseTecnic/nrbookflows/master/lesson1/1-2_weatheralert.json

- Establish an MQTT connection with other devices (Arduino, ESP8266, ESP32 etc);
- Create a responsive graphical user interface for your projects;
- Communicate with third-party services (IFTTT.com, Adafruit.io, ThingSpeak, Home Assistant, InfluxDB etc);
- Retrieve data from the web (weather forecast, stock prices, emails. etc);
- Create time-triggered events;
- Store and retrieve data from a database.

Node-RED

Installation (with npm)

- https://nodered.org/docs/getting-started/
- https://nodered.org/docs/getting-started/local
- https://nodered.org/docs/faq/starting-node-red-on-boot

```
(1) Install Node-RED
     sudo npm install -g --unsafe-perm node-red
(2) Try to run node-red
```

```
$ node-red

Welcome to Node-RED
===================

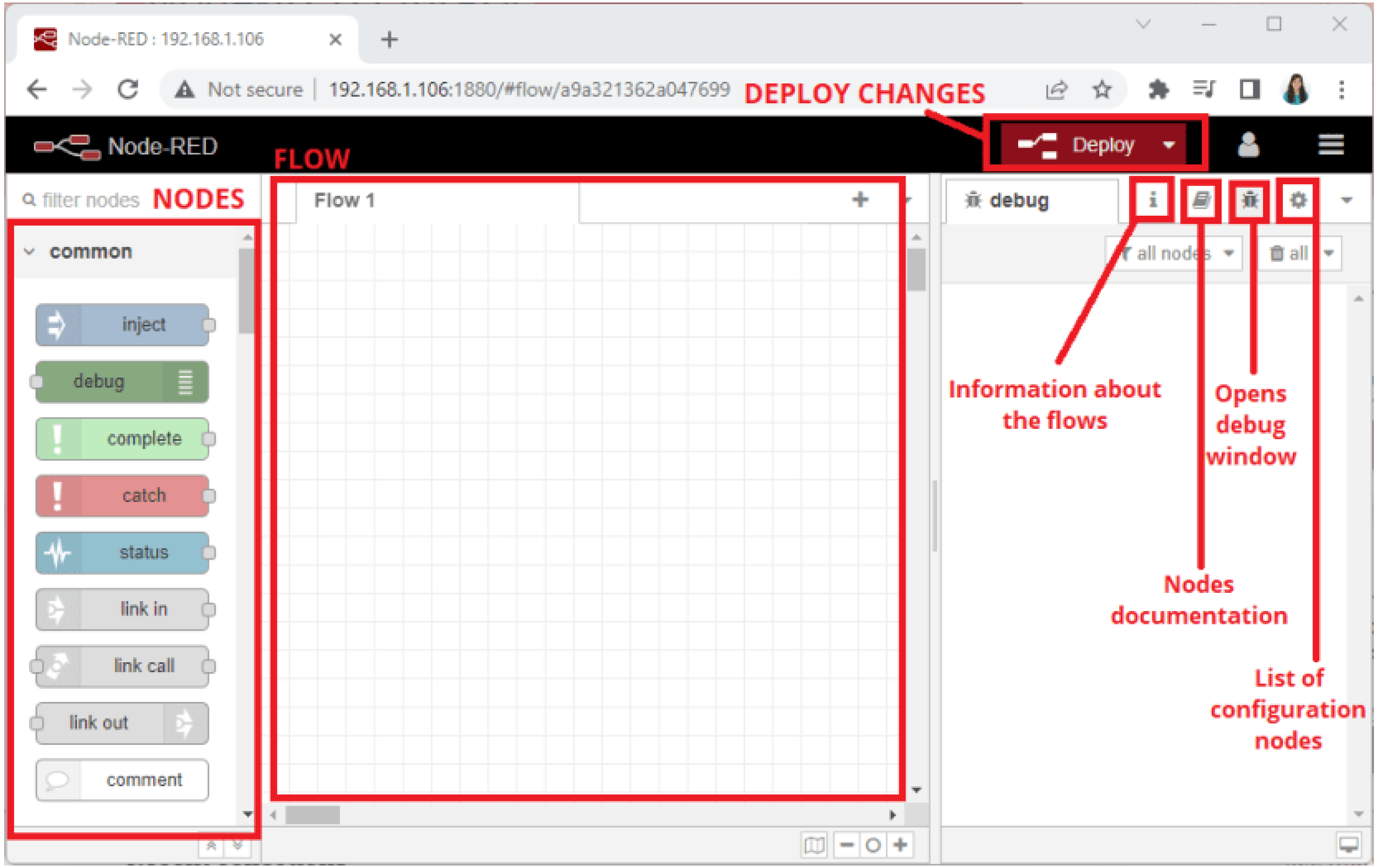
30 Jun 23:43:39 - [info] Node-RED version: v1.3.5
30 Jun 23:43:39 - [info] Node.js  version: v14.7.2
30 Jun 23:43:39 - [info] Darwin 19.6.0 x64 LE
30 Jun 23:43:39 - [info] Loading palette nodes
30 Jun 23:43:44 - [warn] rpi-gpio : Raspberry Pi specific node set inactive
30 Jun 23:43:44 - [info] Settings file  : /Users/nol/.node-red/settings.js
30 Jun 23:43:44 - [info] HTTP Static    : /Users/nol/node-red/web
30 Jun 23:43:44 - [info] Context store  : 'default' [module=localfilesystem]
30 Jun 23:43:44 - [info] User directory : /Users/nol/.node-red
30 Jun 23:43:44 - [warn] Projects disabled : set editorTheme.projects.enabled=true to enable
30 Jun 23:43:44 - [info] Creating new flows file : flows_noltop.json
30 Jun 23:43:44 - [info] Starting flows
30 Jun 23:43:44 - [info] Started flows
30 Jun 23:43:44 - [info] Server now running at http://127.0.0.1:1880/red/
```

(3) Install PM2: process manager for node.js

```
sudo npm install -g pm2
pm2 start /usr/local/bin/node-red -v
```

Note: If you have done a global install of node-red, then on Linux/OS X the node-red command will probably be either: /usr/bin/node-red or /usr/local/bin/node-red. The command which node-red can be used to confirm the location.

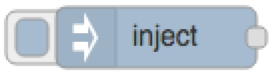
Node-RED Interface Main Sections

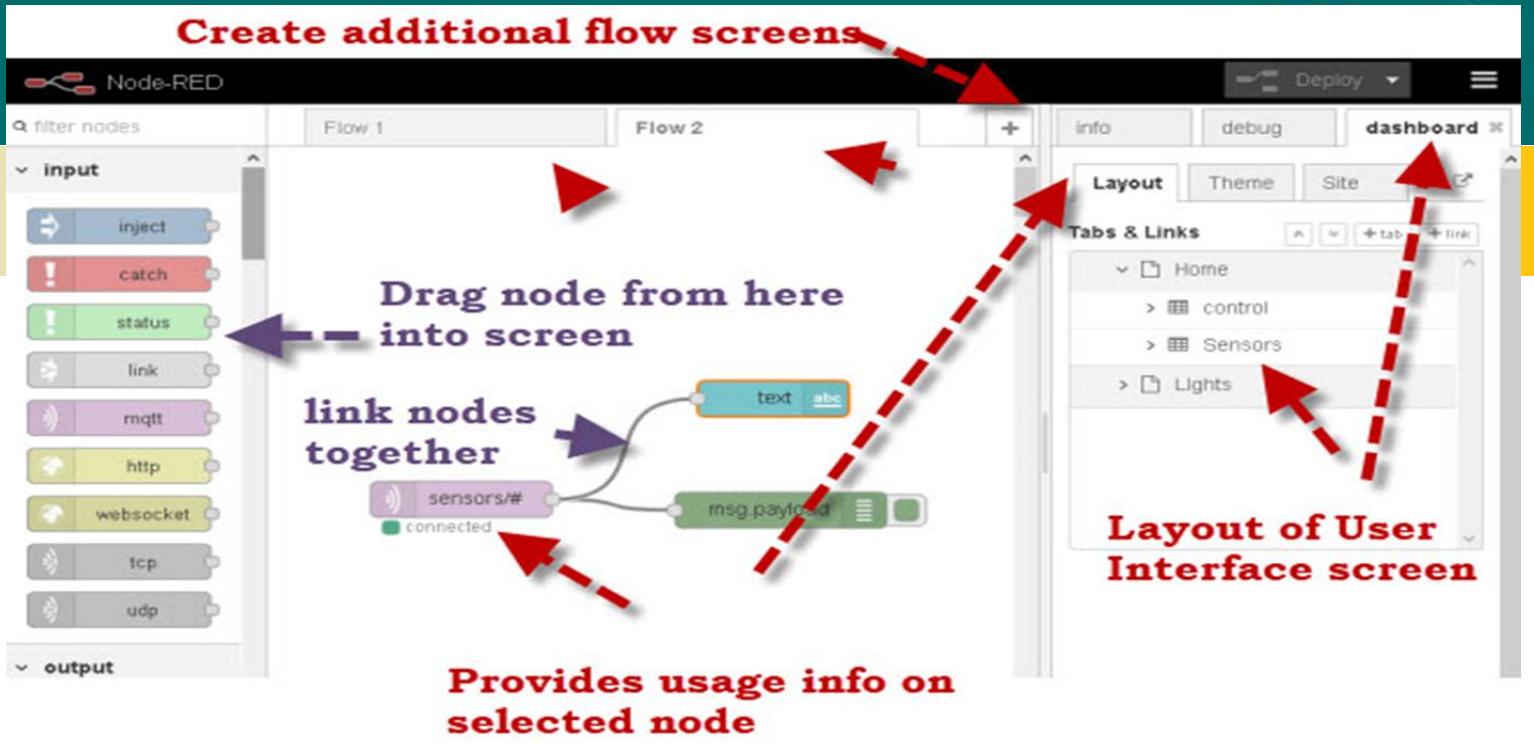


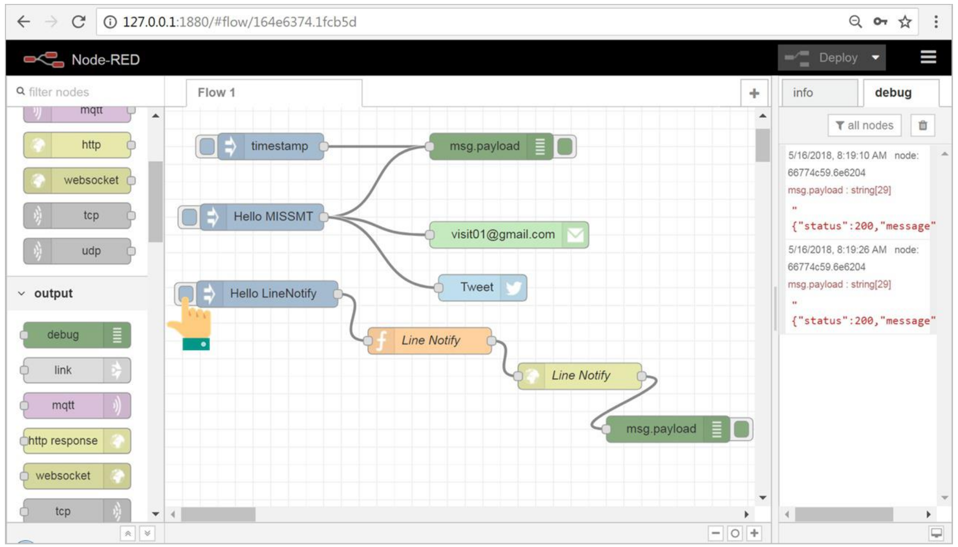
Node-RED

Nodes

- Input Nodes (e.g. inject)
- Processing Nodes (e.g. function)
- Output Nodes (e.g. debug)



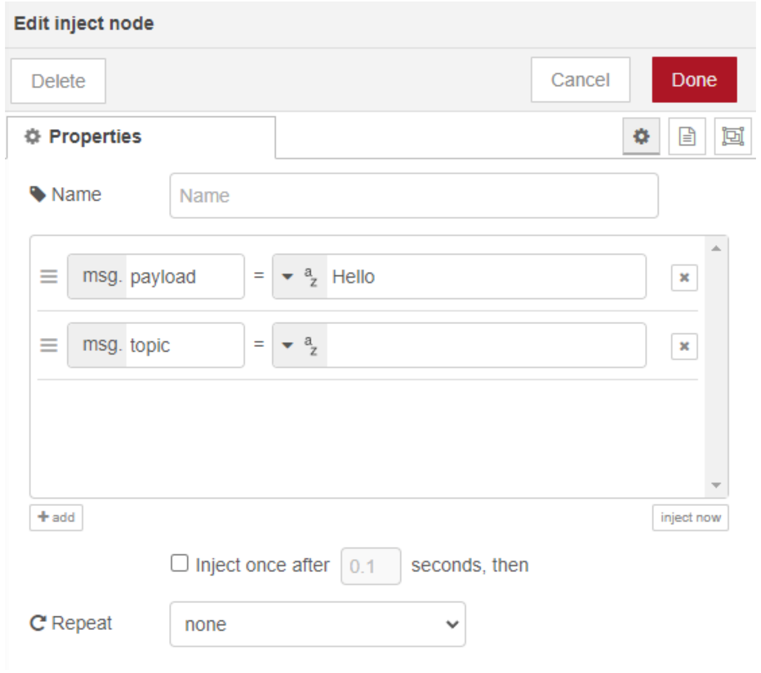




Exercise: Simple Flow

The flow we'll create, simply prints a message to the debug console, when triggered.
Drag an **inject** node and a **debug** node to your flow and wire them together.

Now, let's edit the inject node. Double-click the node. In the figure below, you can see the different settings you can change.
On the `msg.payload` field, select string and type `Hello`. Then, click **Done**.



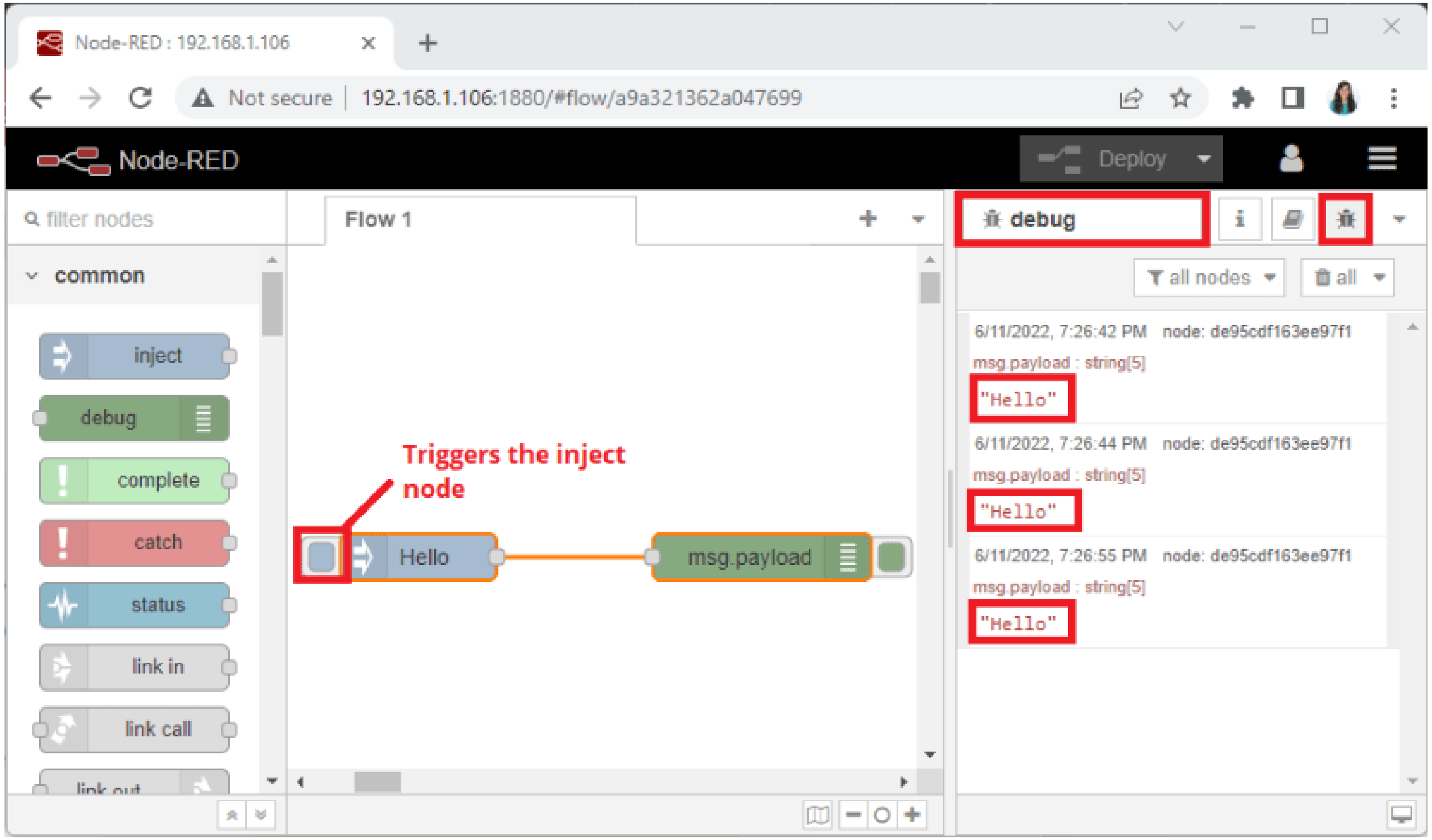
To save your application, you need to click the **Deploy** button in the top right corner.

Your application is saved

Ref: https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/

Exercise: Simple Flow Test

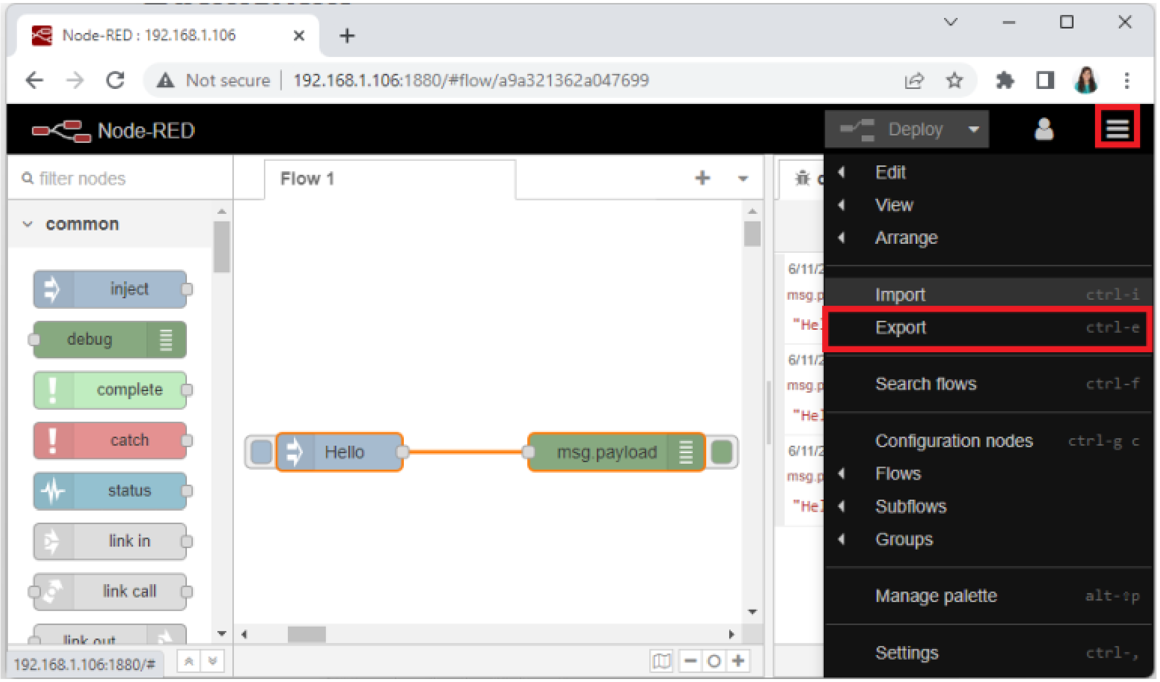
Let's test our simple flow. Open the **debug** window and click the **inject** node to trigger the flow.



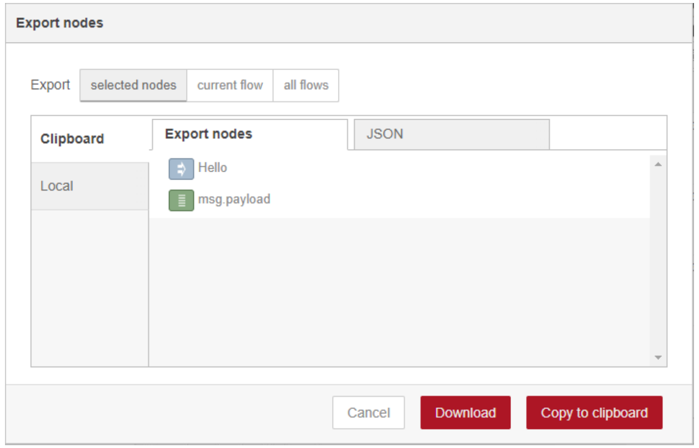
Ref: https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/

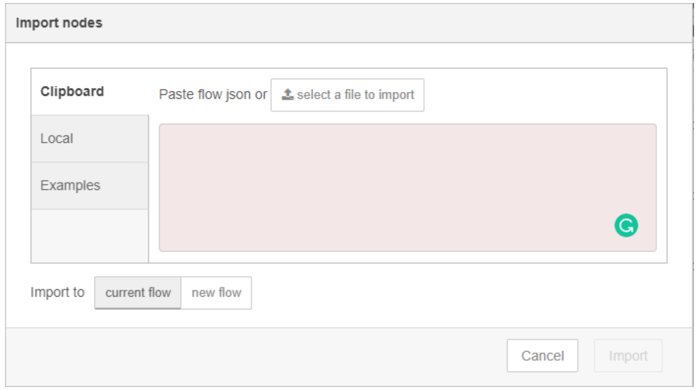
Exporting and Importing Nodes

- Backup your Node-RED flow
- Move your flow to another Node-RED system
- Share your Node-RED project with others



You can select if you want to save the selected nodes, the current flow, or all flows. You can also download the nodes as a JSON file or copy the JSON to the clipboard.

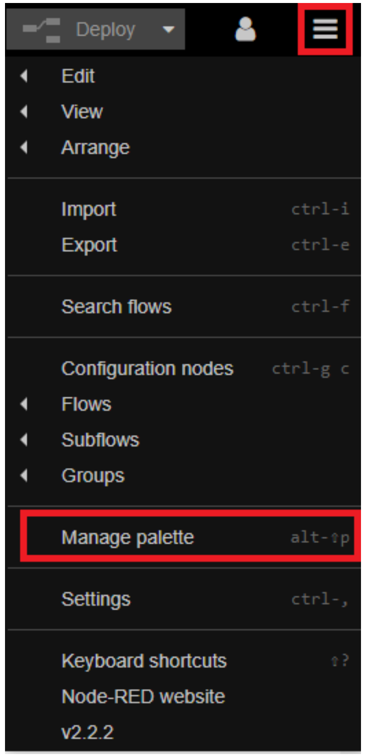


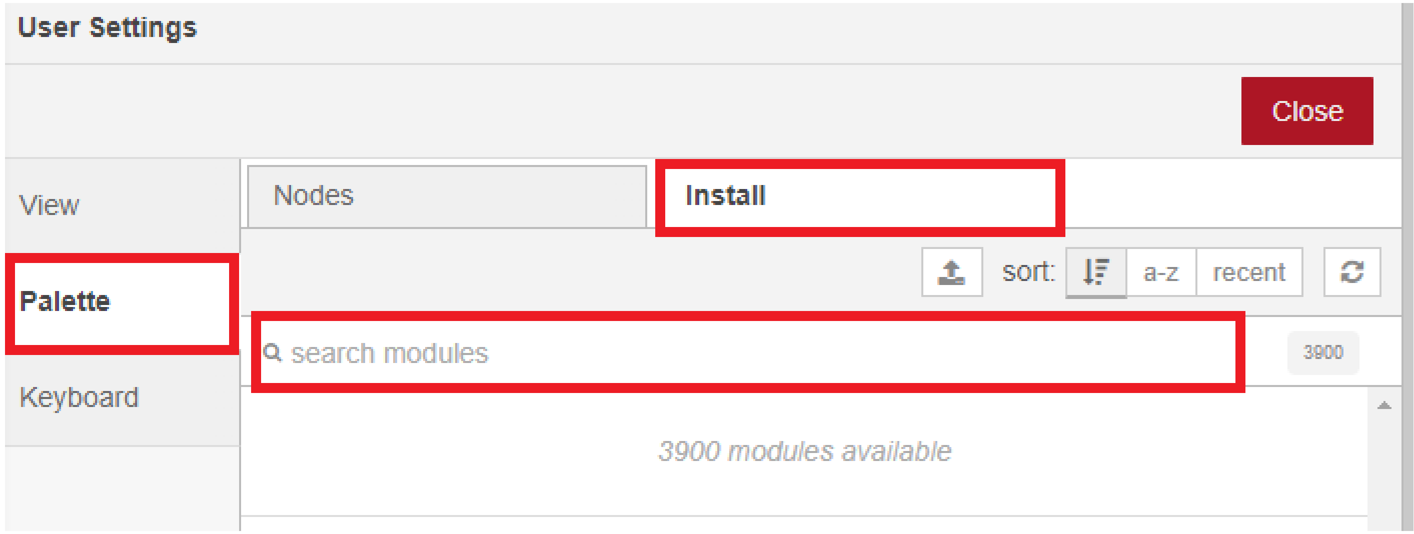


Ref: https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/

Installing Pallete Nodes

- There are many more nodes available that you can install and use for your projects. You can find them in the [Node-RED library](). If you need some specific task for your project, there's probably already a node for that.

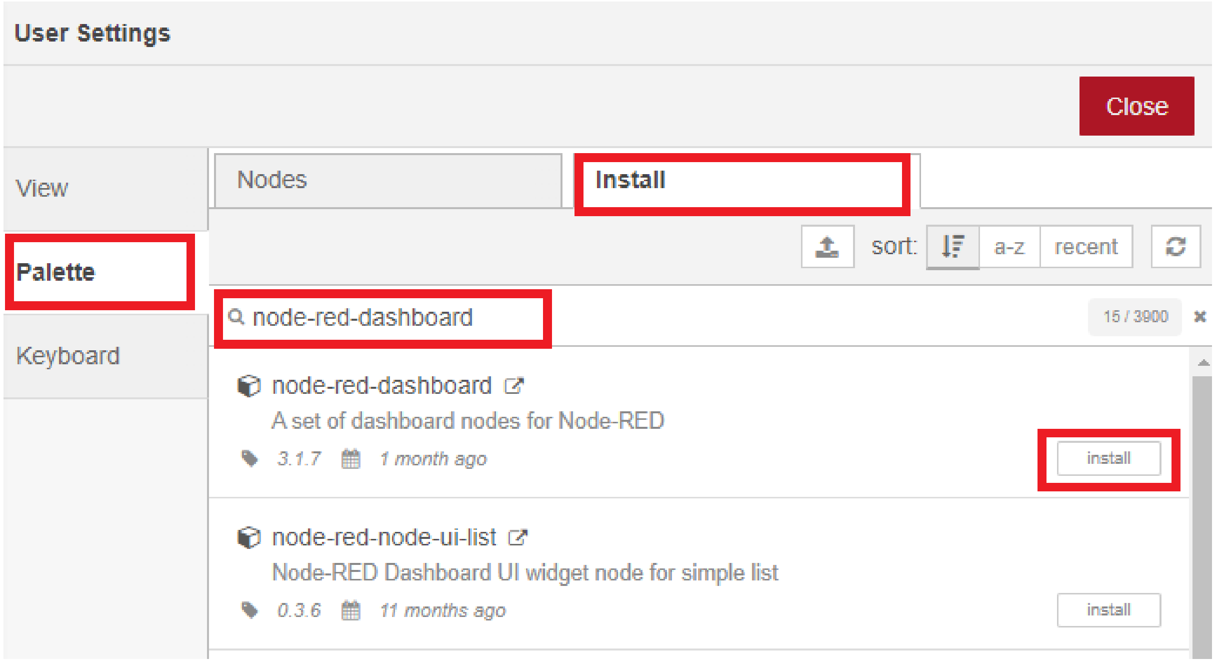


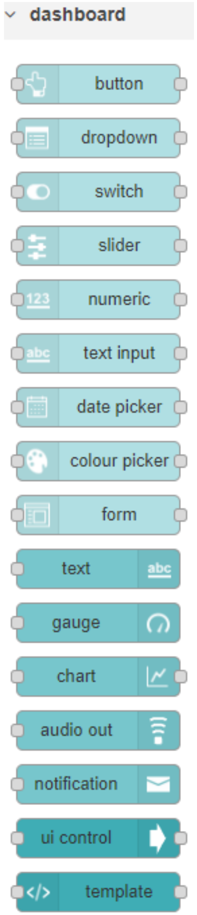


Ref:
https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/
https://flows.nodered.org/

Dashboard

- Node-RED Dashboard is a module that provides a set of nodes in Node-RED to quickly create a live data dashboard. You can install those nodes using the **Menu** > **Manage Palette**. Then, search for node-red-dashboard and install it.





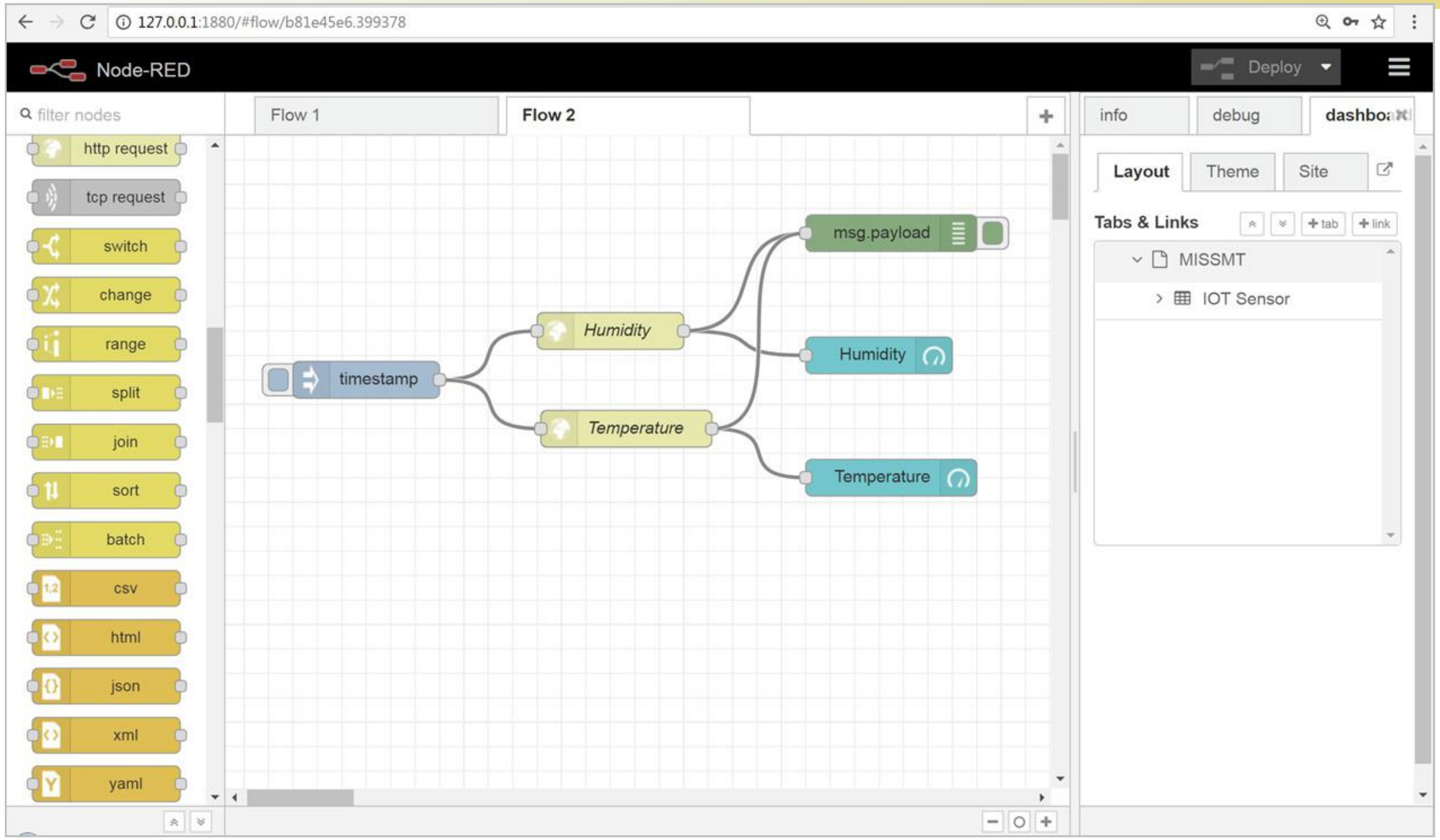
After Installation

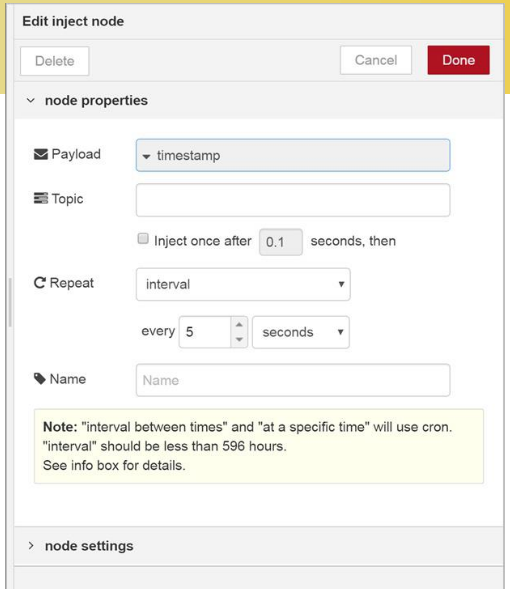
Nodes from the dashboard section provide widgets that show up in your application user interface (UI). The user interface is accessible on the following URL:

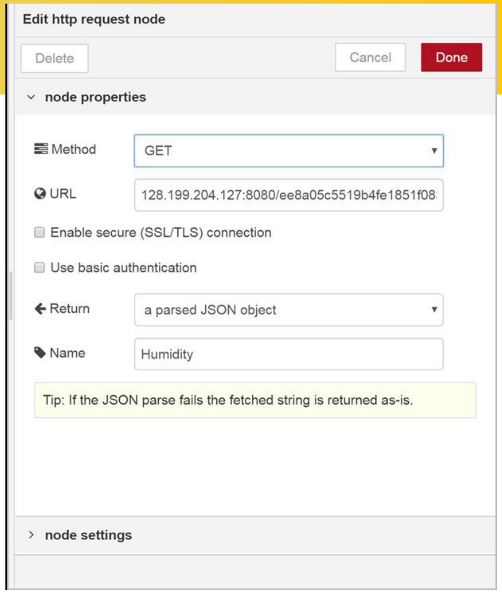
http://Your_NodeRED_IP_address:**1880/ui**

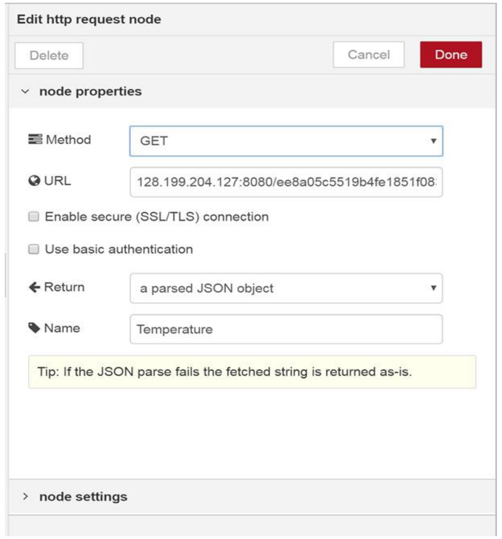
Ref:
https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/
https://flows.nodered.org/

Example (RESTful)

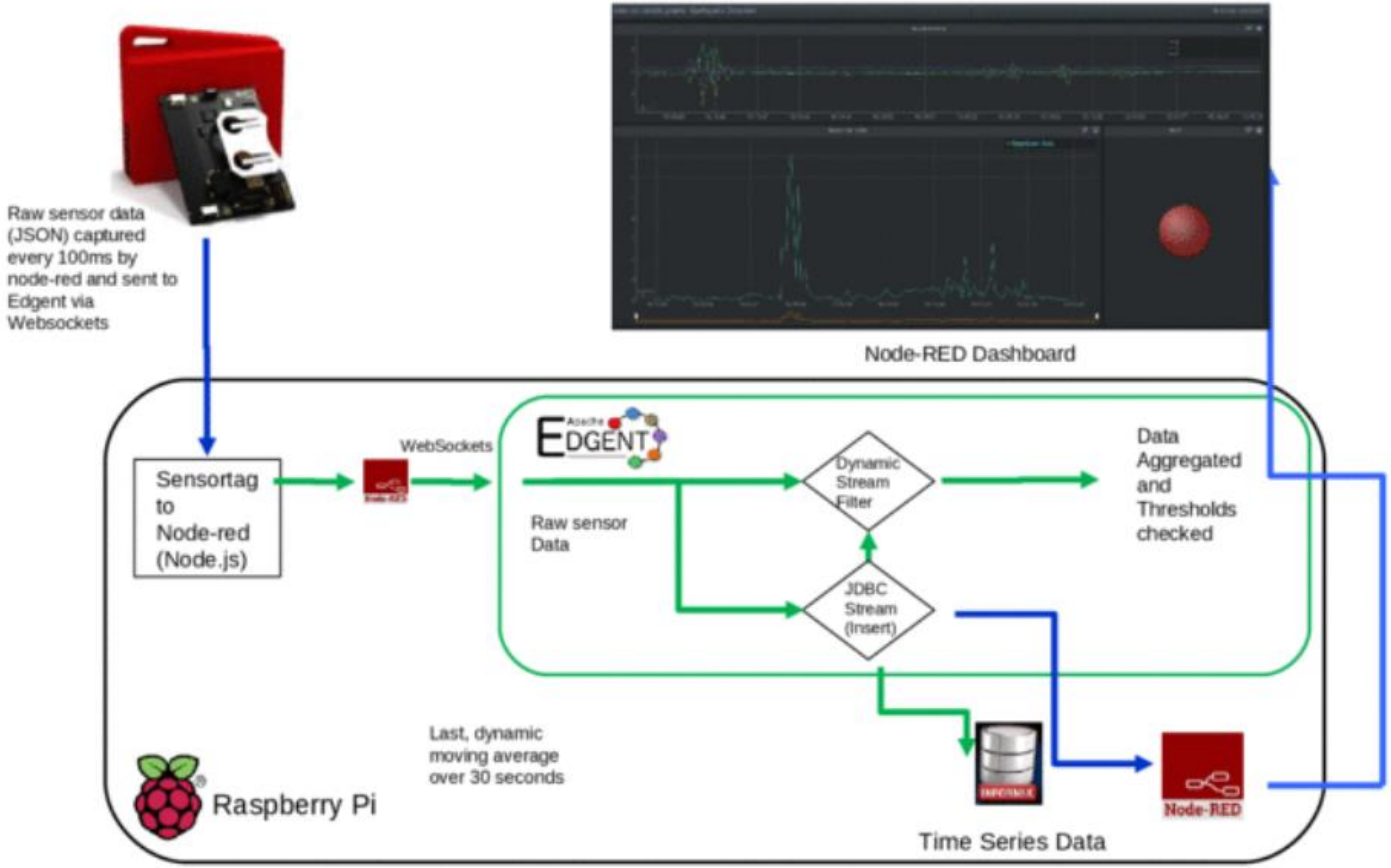






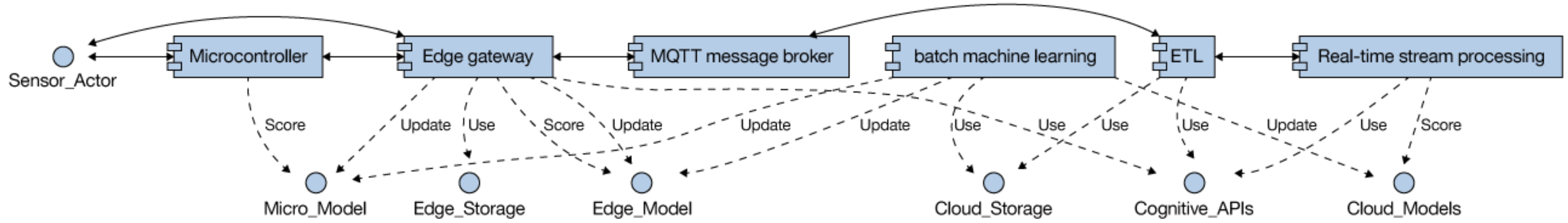


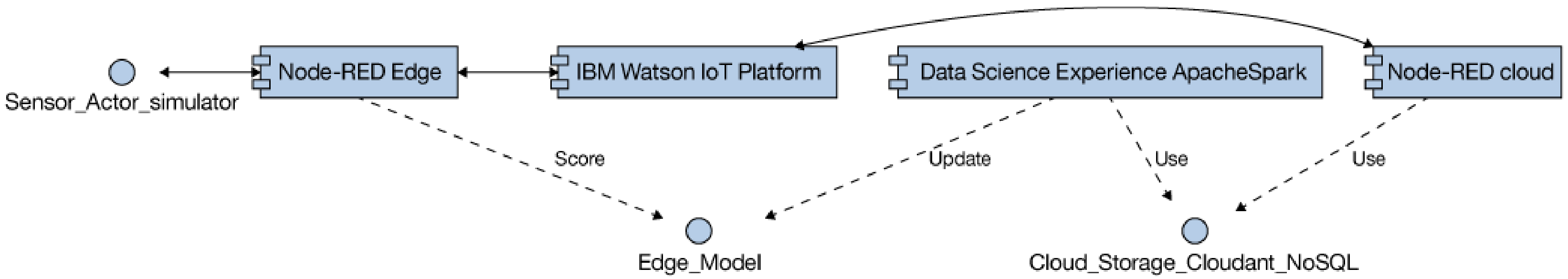
Raspberry Pi



https://vdocuments.site/iot-analytics-from-edge-to-cloud-using-ibm-informix.html

IBM





Component Name	Component Type	UML Stereotype	Component Description
Node-RED Edge	Node-RED	Edge Gateway	Used to create data flow application. Node-RED is an open source data flow editor written in JavaScript and running on Node.js. IBM created it and donated it to the JavaScript foundation.
Sensor_Actor_simulator	Node-RED node	Sensor or Actor	Used to simulate a sensor or actor in the absence of a physical IoT system.
Watson Studio	Apache Spark and Jupyter notebooks as a service	Batch Machine Learning	Used to detect anomalies in real time on an IoT sensor time-series stream.
IBM Watson IoT Platform	IBM Watson IoT Platform	MQTT message broker	Acts as asynchronous glue between all components in the IoT operational model.
Node-RED Cloud	Node-RED	ETL plus real-time stream processing	Used for streaming IoT sensor data to cloud storage.
Cloud_Storage_Cloudant_NoSQL	Cloudant	Cloud_Storage	Used to store IoT sensor data. Cloudant is an Apache CouchDB as a service. We can also use SQL databases or OpenStack Swift Object Storage (which is the most cost-effective option).
Edge_Model	Node-RED	Edge_Model	Holds a simple threshold value that gets populated by the Batch Machine Learning component dynamically.

https://developer.ibm.com/technologies/iot/tutorials/iot-cognitive-iot-app-machine-learning/

Operational Model Components

Component Name	Component Type	UML Stereotype	Component Description
Node-RED Edge	Node-RED	Edge Gateway	Used to create data flow application. Node-RED is an open source data flow editor written in JavaScript and running on Node.js. IBM created it and donated it to the JavaScript foundation.
Sensor_Actor_simulator	Node-RED node	Sensor or Actor	Used to simulate a sensor or actor in the absence of a physical IoT system.
Watson Studio	Apache Spark and Jupyter notebooks as a service	Batch Machine Learning	Used to detect anomalies in real time on an IoT sensor time-series stream.
IBM Watson IoT Platform	IBM Watson IoT Platform	MQTT message broker	Acts as asynchronous glue between all components in the IoT operational model.
Node-RED Cloud	Node-RED	ETL plus real-time stream processing	Used for streaming IoT sensor data to cloud storage.
Cloud_Storage_Cloudant_NoSQL	Cloudant	Cloud_Storage	Used to store IoT sensor data. Cloudant is an Apache CouchDB as a service. We can also use SQL databases or OpenStack Swift Object Storage (which is the most cost-effective option).
Edge_Model	Node-RED	Edge_Model	Holds a simple threshold value that gets populated by the Batch Machine Learning component dynamically.

https://developer.ibm.com/technologies/iot/tutorials/iot-cognitive-iot-app-machine-learning/

References & Tutorials

https://nodered.org/
https://flows.nodered.org/

https://developer.ibm.com/learningpaths/get-started-node-red/

https://stevesnoderedguide.com/node-red-dashboard

https://randomnerdtutorials.com/getting-started-node-red-dashboard/

https://www.youtube.com/results?search_query=node-red
- Node-RED Essential: https://www.youtube.com/watch?v=ksGeUD26Mw0&list=PLyNBB9VCLmo1hyO-4fIZ08gqFcXBkHy-6
- Understanding Node-RED: https://www.youtube.com/watch?v=raV5NFInPio

Internet of Things & NETPIE2020 – Workshop

(For 29 Oct'21)

Mr. Piyawat Jomsathan (Tae)
Cyber-Physical Systems (CPS)
National Electronics and Computer Technology Center (NECTEC), Thailand

---

NSTDA Anniversary
NECTEC a member of NSTDA

Download Document & Software Link --> https://bit.ly/3BXJzeh

Main Topic

1. What is IoT Platform and NETPIE ?
2. Structure of NETPIE2020
3. Type of Communication in NETPIE2020
4. Data Management in NETPIE2020
5. Freeboard in NETPIE2020
6. Workshop

1

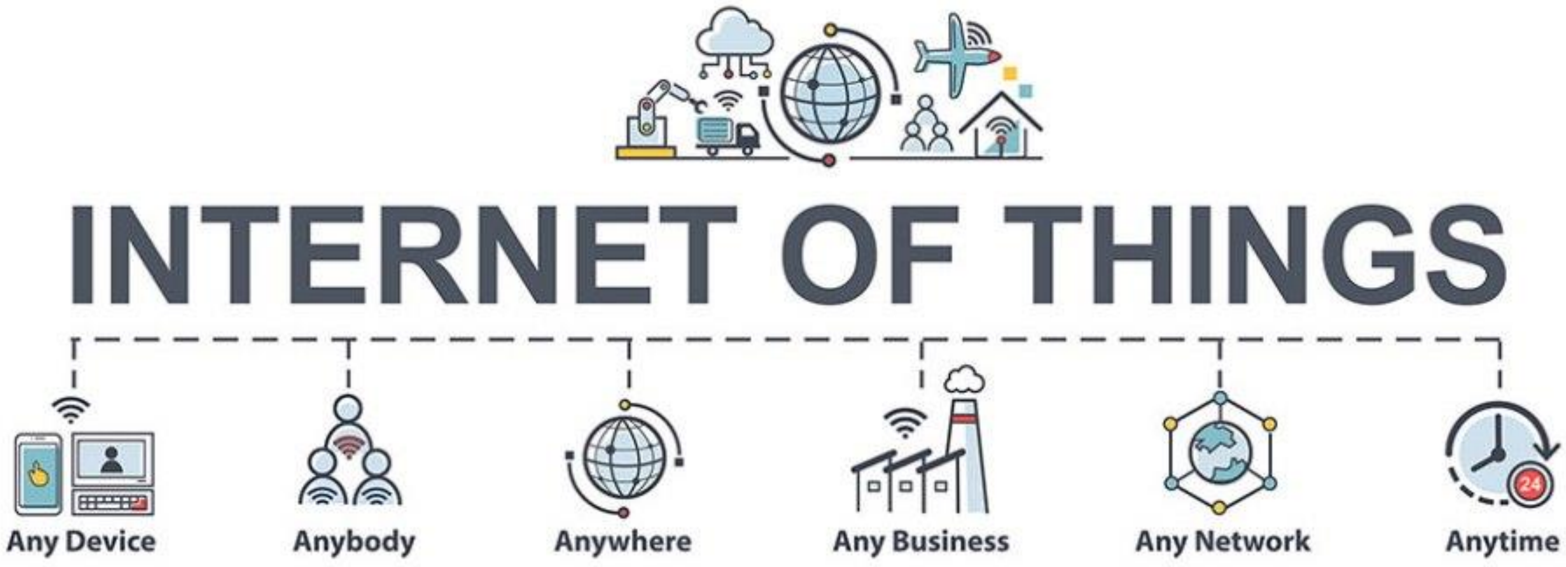
What is IoT Platform and NETPIE ?

NETPIE
where things chat

สวทช NSTDA 30th Anniversary

NECTEC a member of NSTDA

1 - What is IoT Platform and NETPIE ?

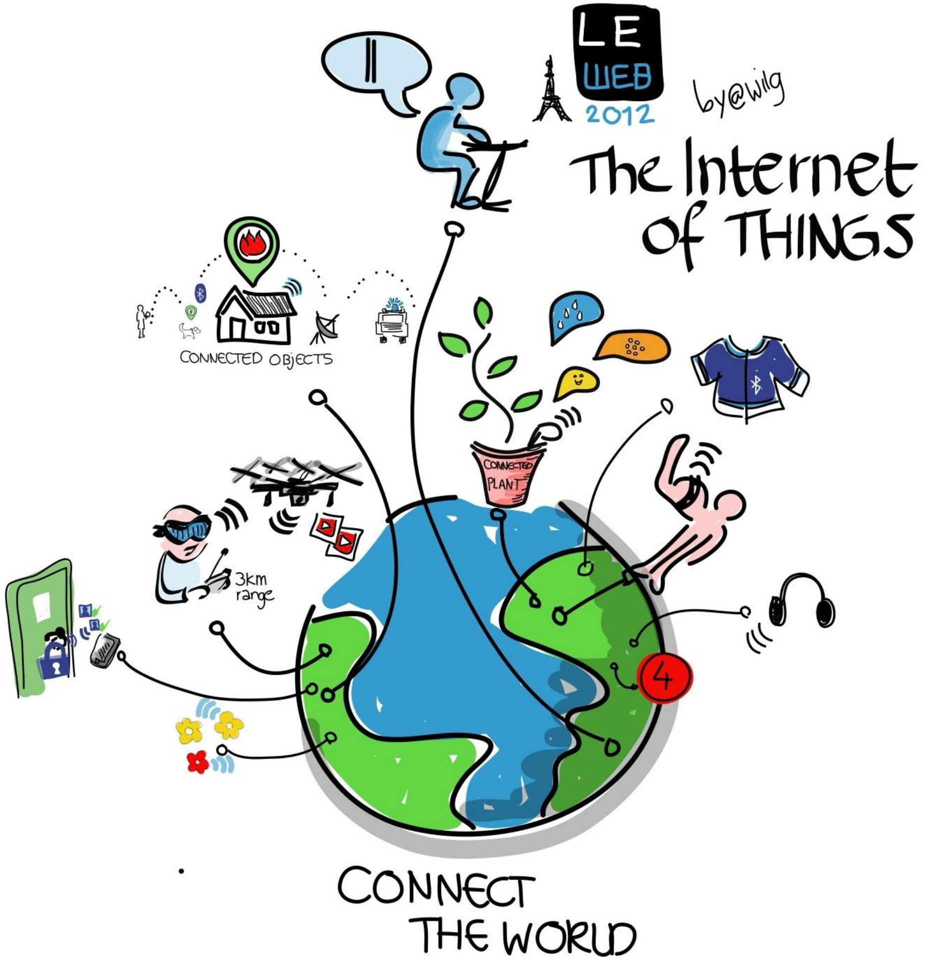


What is IoT??

1 - What is IoT Platform and NETPIE ?

What is IoT??

IoT (Internet of Things) refers to how things are connected to the Internet, enabling humans to control or command various devices via the Internet, such as turning electrical equipments on/off, operating communication devices, modern office, agricultural and industrial machines, and other appliances in our daily life.



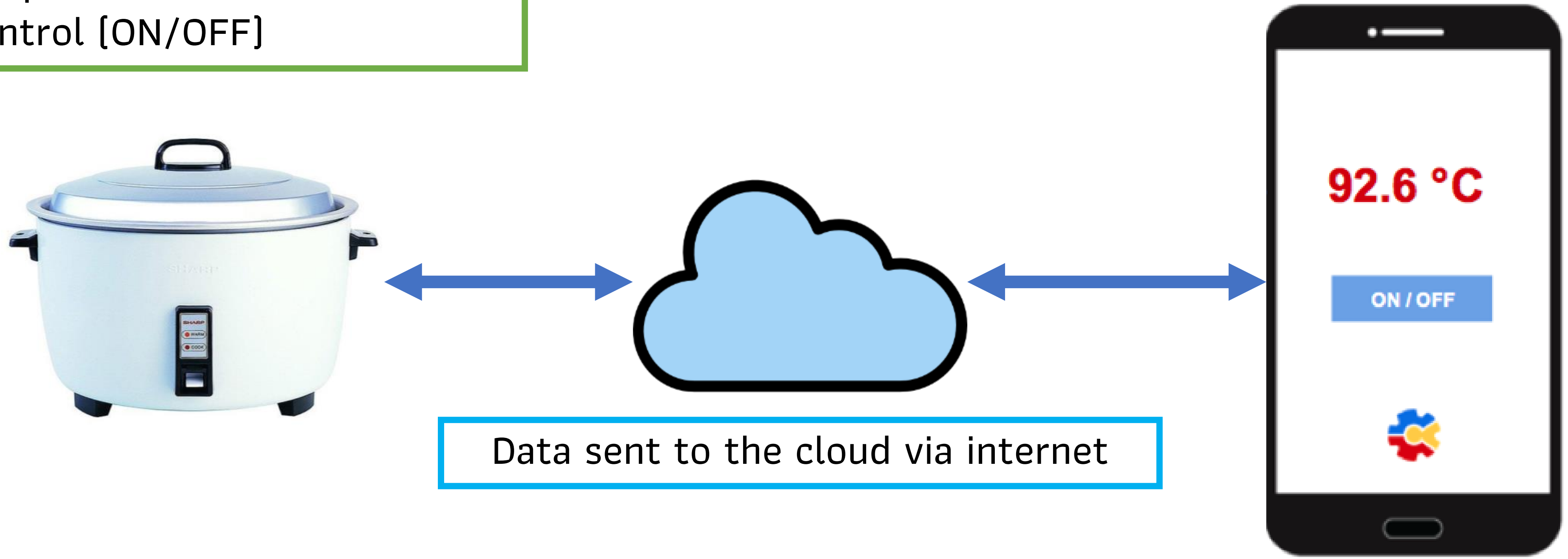
What is <u>IoT Platform</u>??

1 - What is IoT Platform and NETPIE ?

Simple IoT example

What we want from a rice cooker
1. temperature
2. Control [ON/OFF]

Temperature can be monitored via Mobile phone

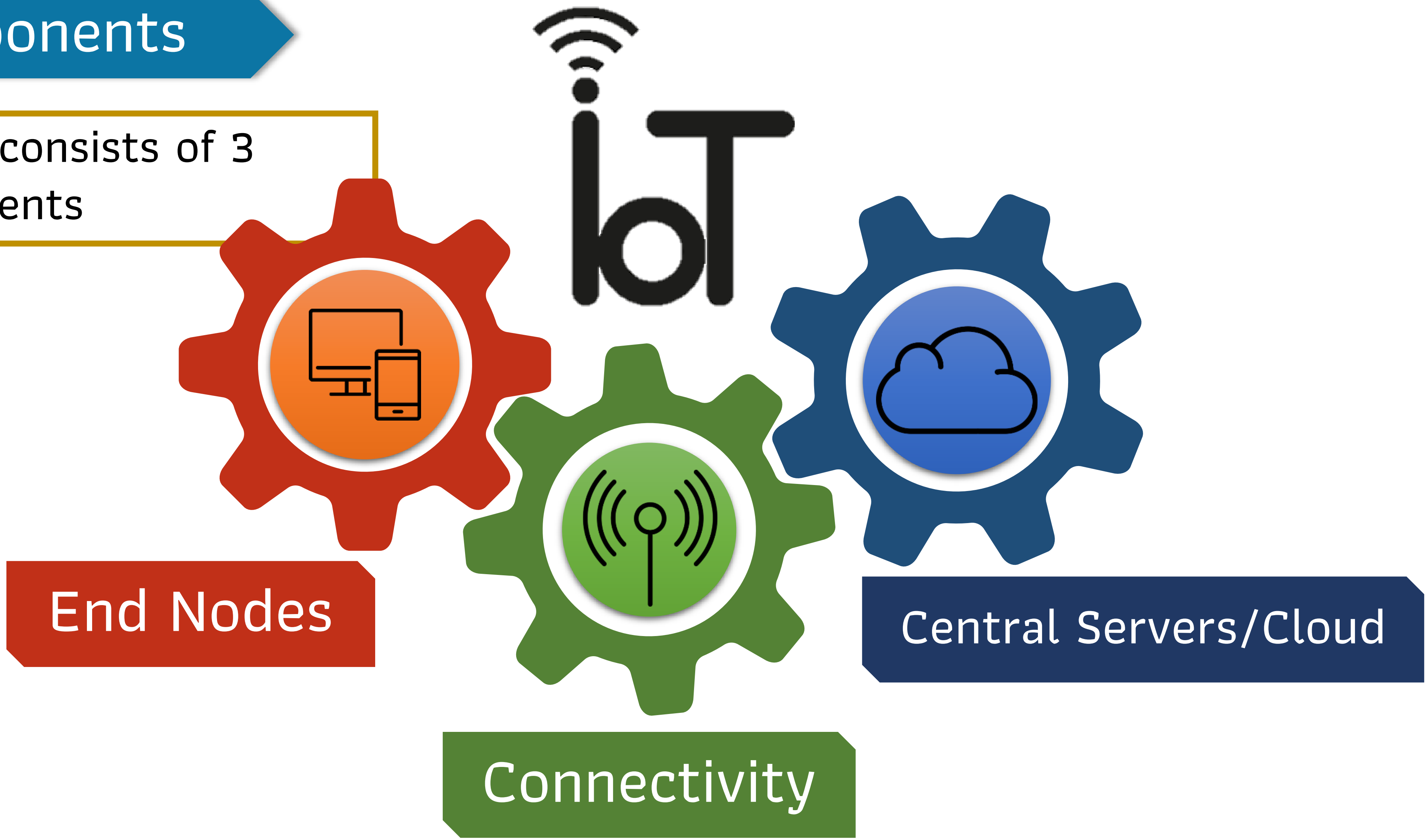


Data sent to the cloud via internet

1 - What is IoT Platform and NETPIE ?

IoT Components

An IoT system consists of 3 important components



1 - What is IoT Platform and NETPIE ?

IoT components

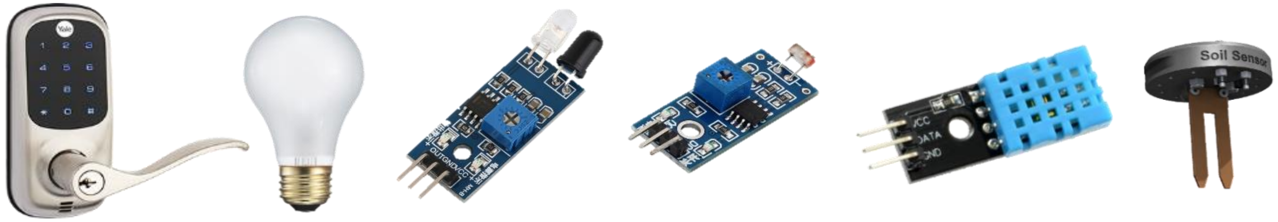
End Nodes

End nodes are parts of objects (bulbs, motors, sensors) and devices that are used to control things or detect and measure parameters of interest such as temperature, humidity, lighting, etc. What are the things for IoT? However, it is only necessary to install an embedded device in order to connect to the Internet. With the currently popular devices including Microcontroller, Single Board, etc.

Microcontrollers

Single Board

Actuators and Sensors



1 - What is IoT Platform and NETPIE ?

IoT Components

Connectivity

Connectivity is a system that allows devices and things to connect to the Internet. To send and receive data between devices to the Cloud, there are various types of Internet connection systems, either wired or wireless depending on the suitability of use.

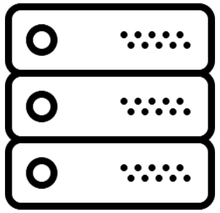
1 - What is IoT Platform and NETPIE ?

IoT Components

Central Servers/Cloud

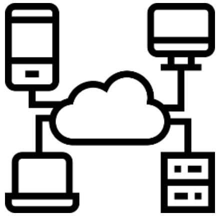
Central Servers / Cloud is a storage area to be used to create a Web Server, Application Server, etc. At present, for IoT systems, the cloud is used as a storage area. The differences between Server and Cloud are

Server



Server is a machine or computer program that provides services in the network to clients. The computers serving as this server should be high-performance, stable, able to serve a large number of users.

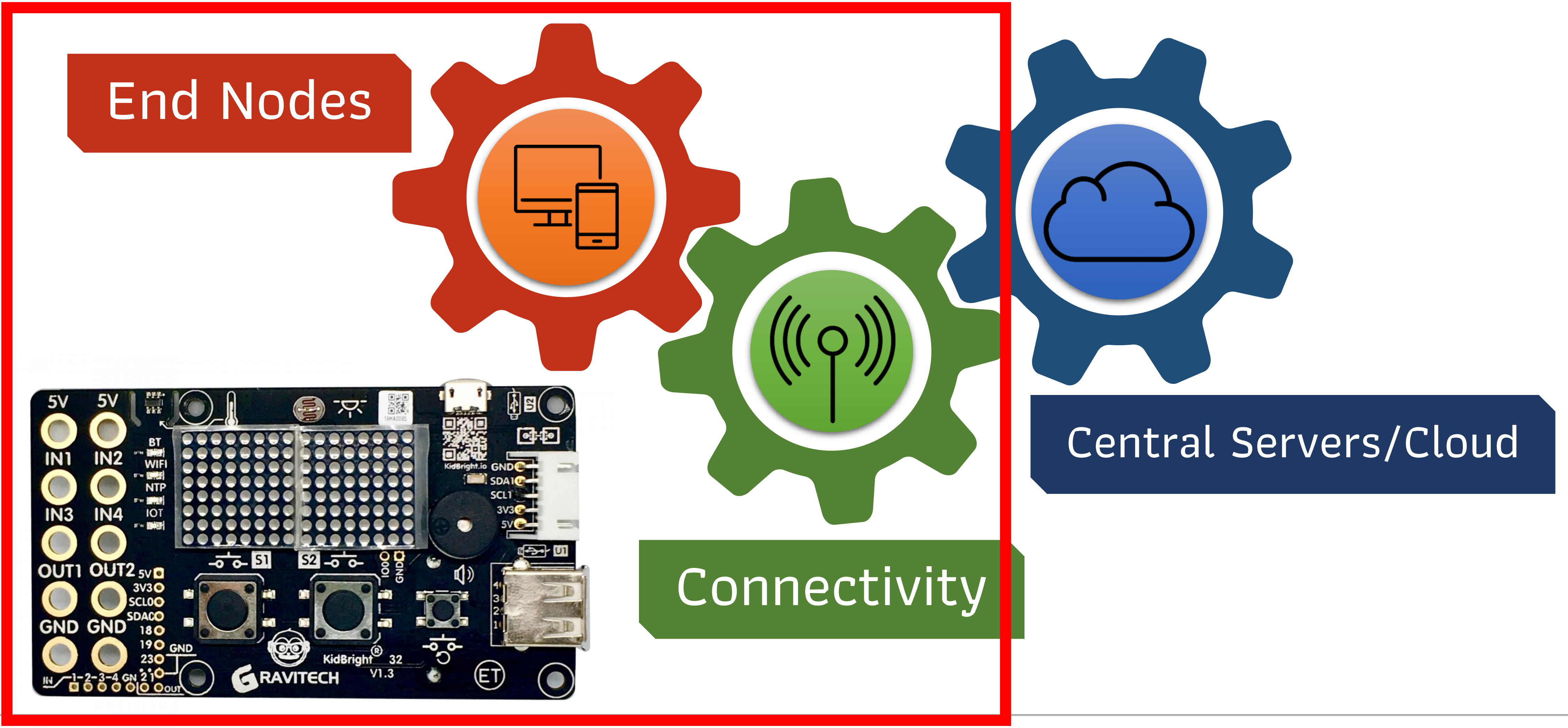
Cloud



Cloud is to bring multiple servers to work together with a high level of computing power. Cloud can build services to perform a variety of tasks.

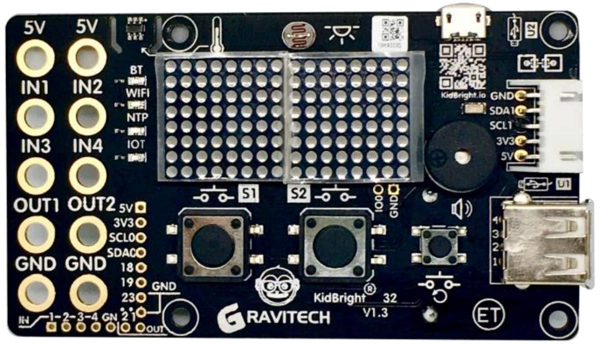
1 - What is IoT Platform and NETPIE ?

IoT Components



1 - What is IoT Platform and NETPIE ?

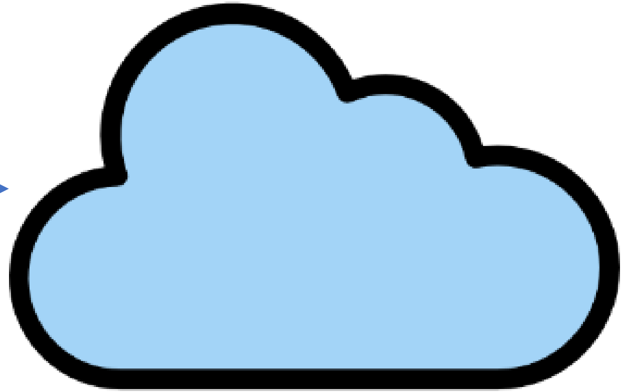
IoT Components



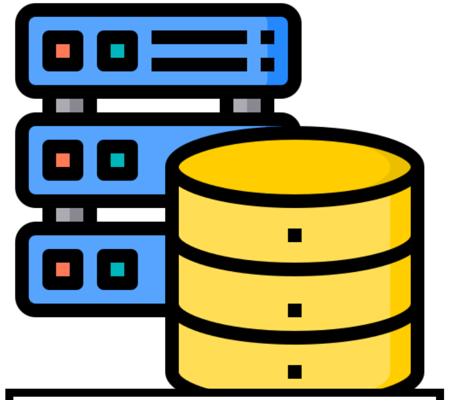
KidBright

Data that KidBright wants to send to Cloud
"Temperature" = 25 °C
"Light" = 80 Lux
"LED Status" = "OFF"

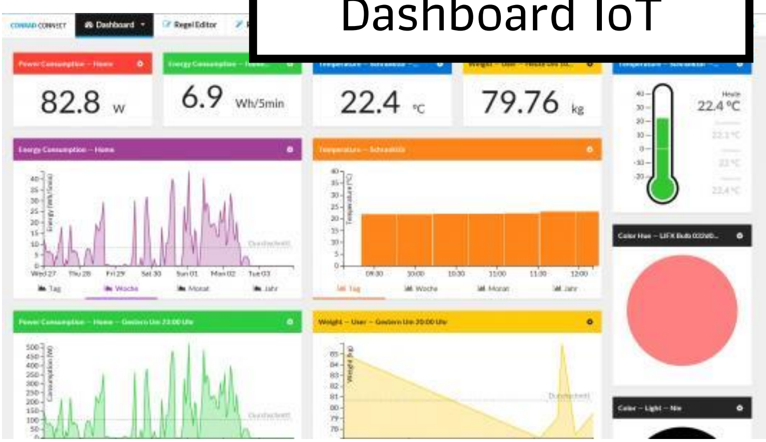
Way of transmitting data / path of transmission



Cloud



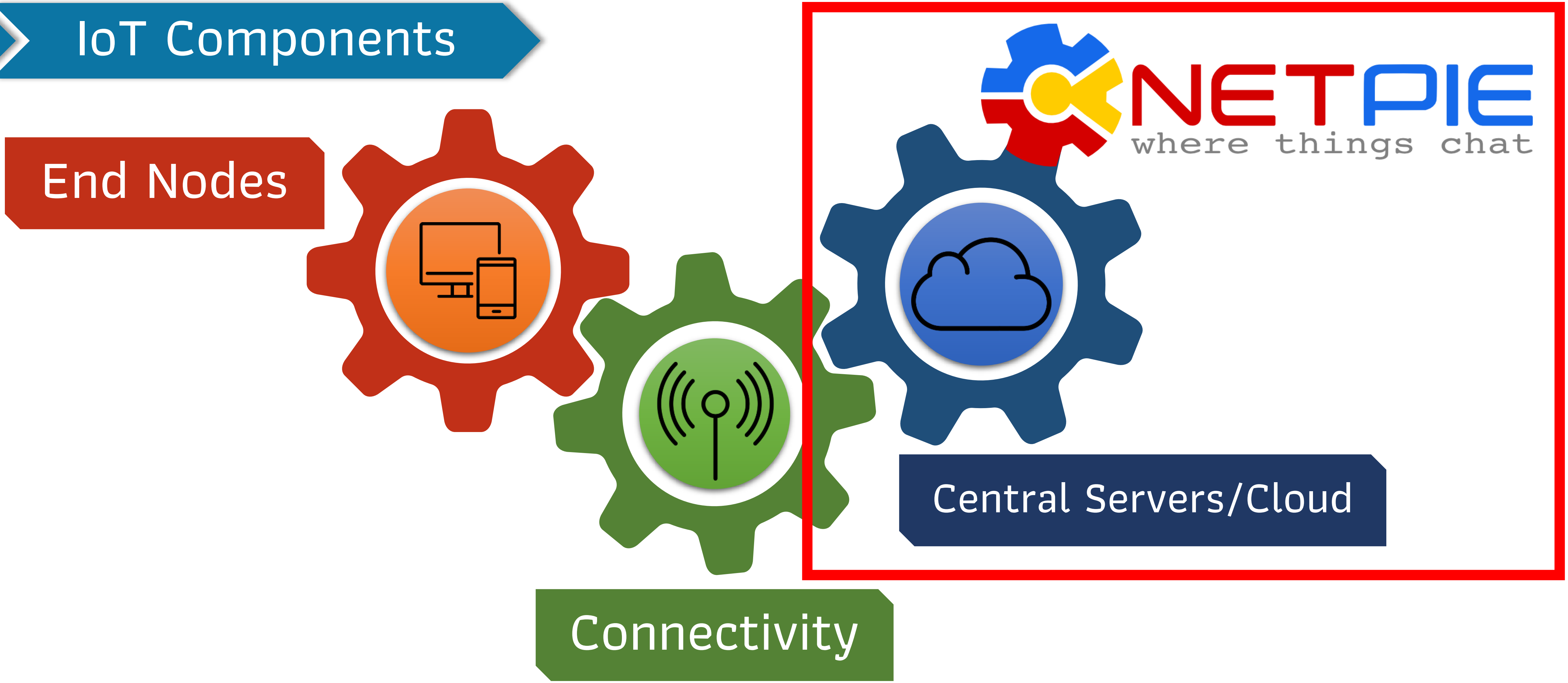
Data Storage



Dashboard IoT

1 - What is IoT Platform and NETPIE ?

IoT Components



1 - What is IoT Platform and NETPIE ?

NETPIE
where things chat

What is NETPIE?

NETPIE is an IoT cloud platform that is open to the public. The platform will help devices able to communicate with each other, receive - transmit data between devices in real-time, allowing users to know the information of the device at that time, no matter where the user is.

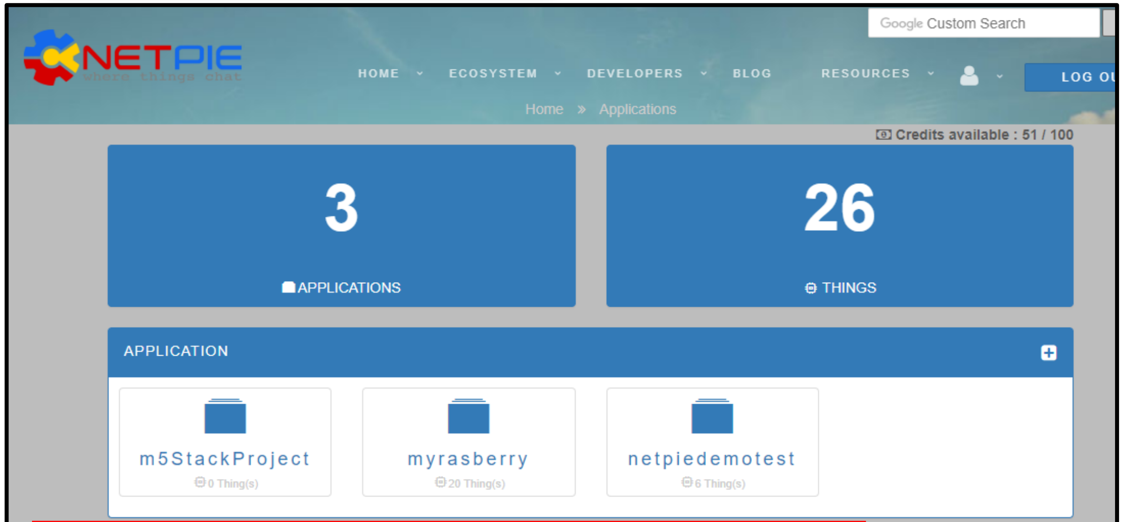
Main features of NETPIE

1. Monitoring : to display device or sensor data in real time
2. Controlling : to control the operation of devices via the cloud platform
3. Data Storage : to collect data from sensors or devices
4. Notification : to alert the user when sensor malfunction occurs

1 - What is IoT Platform and NETPIE ?

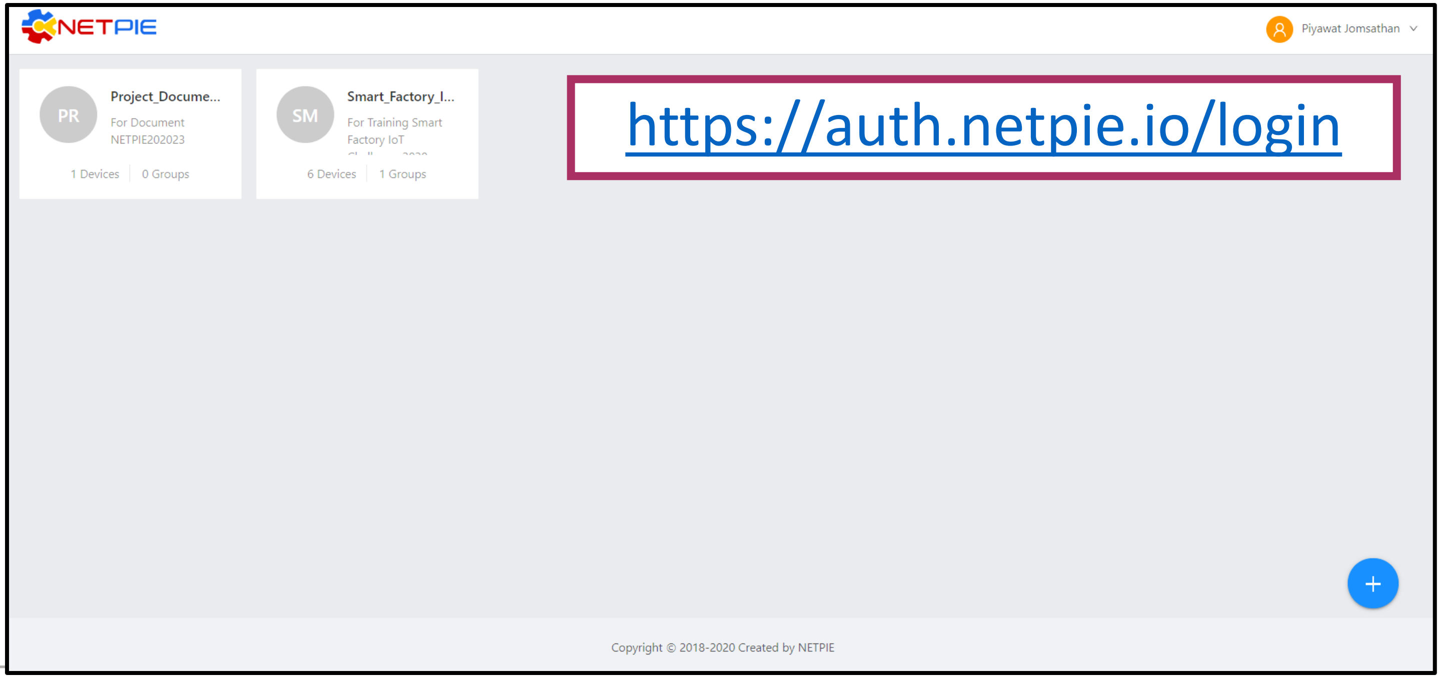
At present, NETPIE is developed in 2 versions
NETPIE 2015 and NETPIE 2020

NETPIE 2015



https://2015.netpie.io/login

NETPIE 2020



https://auth.netpie.io/login

1 - What is IoT Platform and NETPIE ?

NETPIE 2020 and NETPIE 2015 Comparison

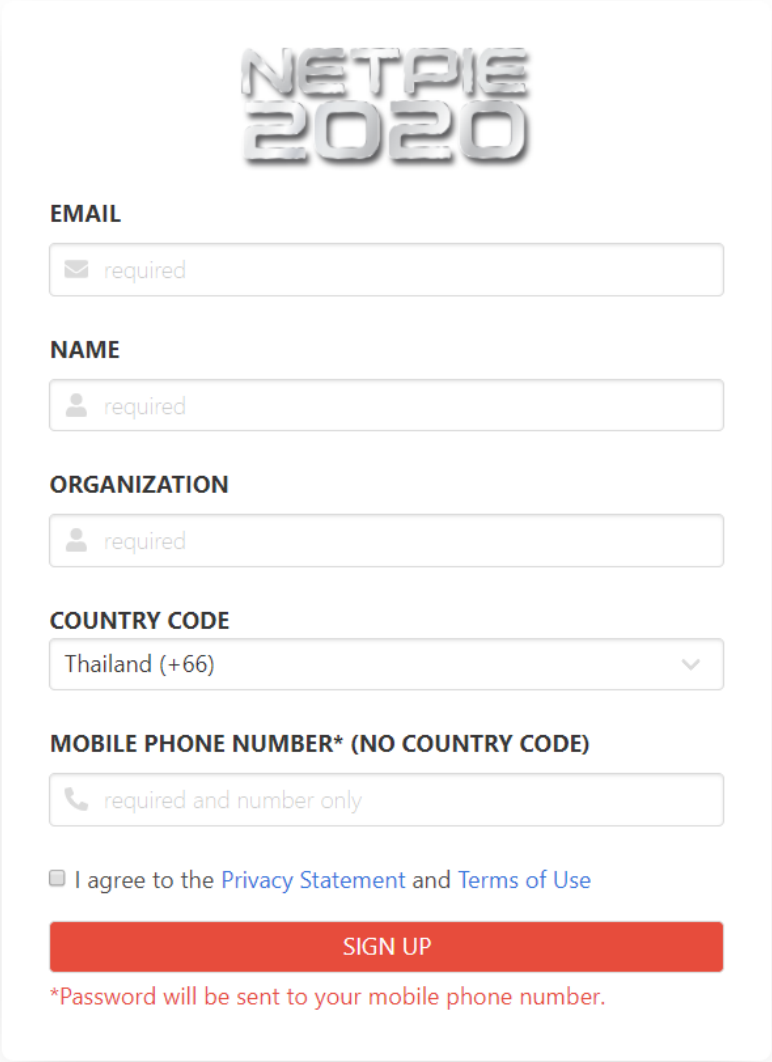
	NETPIE 2020	NETPIE 2015
Design Philosophy	Platform - Centric	Device – Centric
Commercial Ability	Commercial – Ready	Need to re-program Hardware migrate to commercial platform
Suitable Usage 1	Mass production, Project-based	Project-based
Target User	IoT consumer product makers, Hobbyists, Students	Makers, hobbyists, students
Communication Protocol	MQTT, HTTP	Microgear
Programming Language	Any languages with MQTT library support	Limited to Microgear library
Hardware Support 2	Unlimited as long as it supports MQTT	Limited to those with Microgear support
Device Identity and Group	No APPID Device identity and group can be adjusted after product is sold/installed	Use APPID Device identity and group must be programmed into firmware
Rate Limit	Allow burst	Everyone is subject to the same rate limit
Trigger 3	Can set trigger action in cloud platform	Set trigger action inside IoT devices

2 Structure of NETPIE2020

2 - Structure of NETPIE2020

Getting started with NETPIE2020

Create NETPIE2020 user account at
https://auth.netpie.io/signup

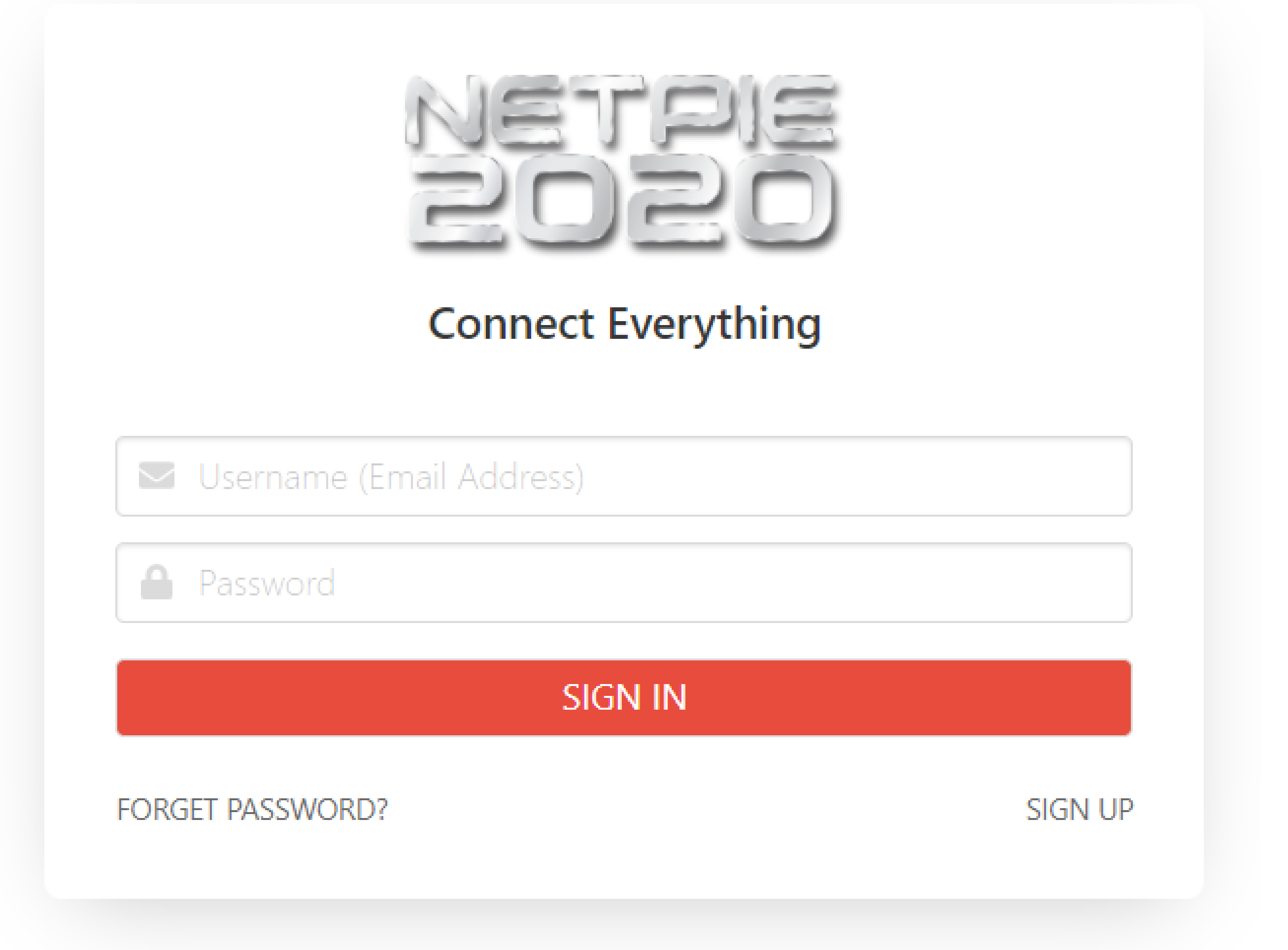


2 - Structure of NETPIE2020

Getting started with NETPIE2020

After sign up NETPIE2020, login
With Username (email) and Password

Or login at this link
https://auth.netpie.io/login



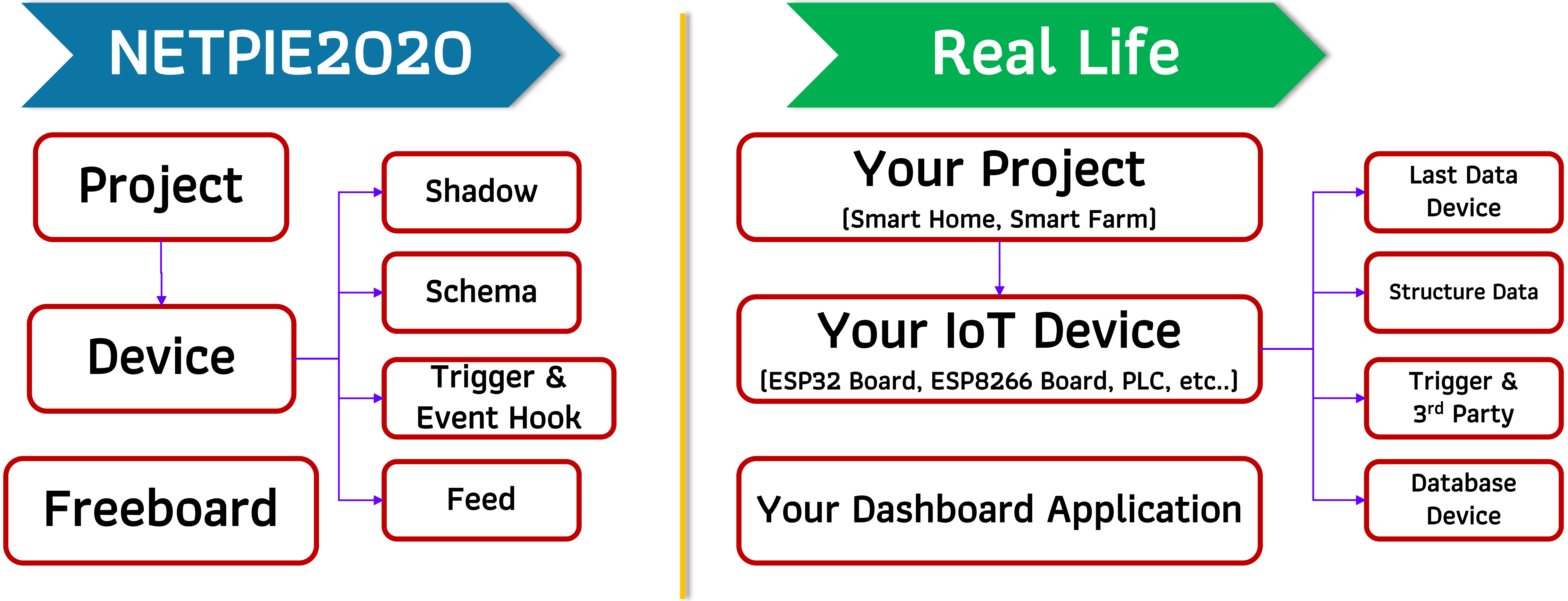
2 - Structure of NETPIE2020

Getting started with NETPIE2020

After login, you will see the window as shown

2 - Structure of NETPIE2020

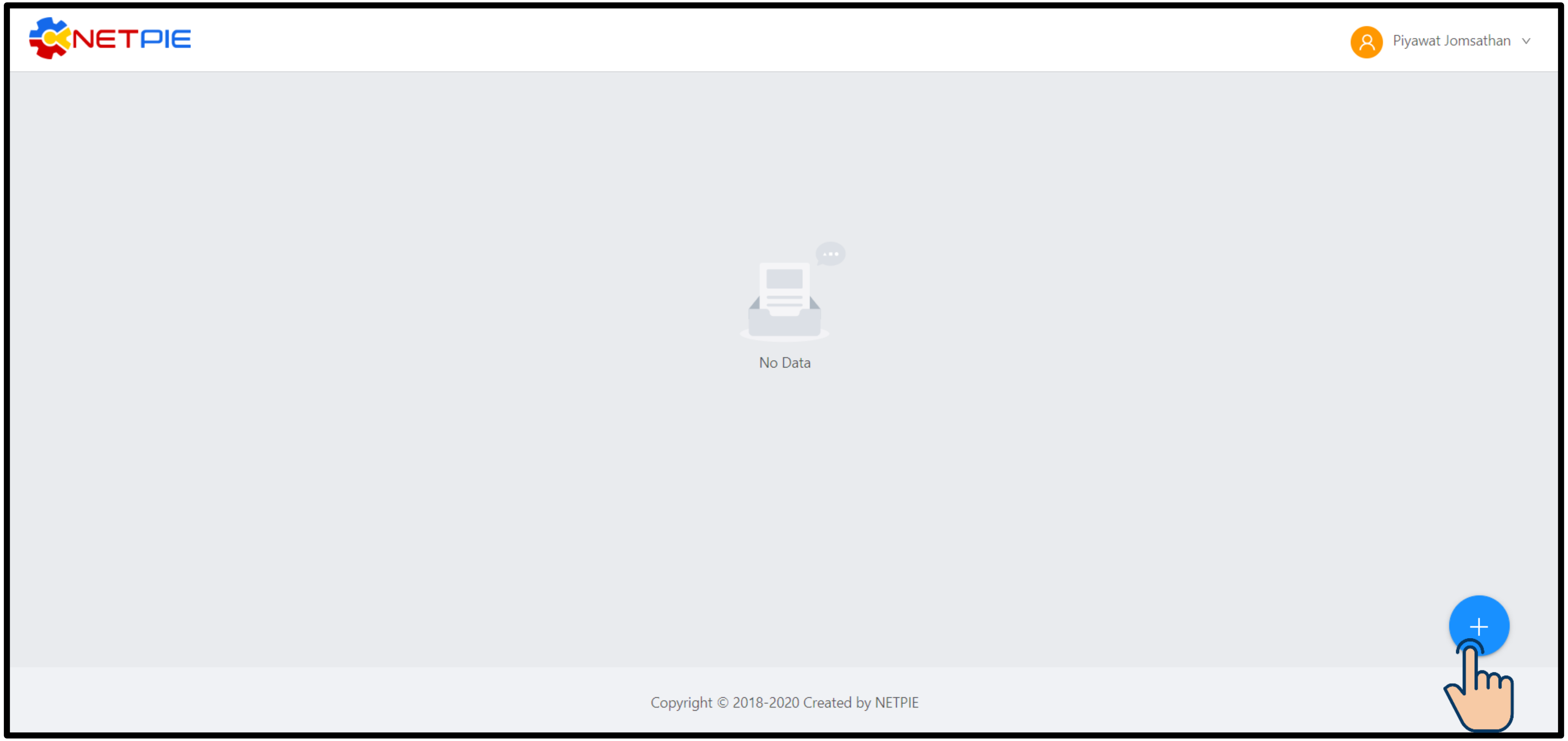
Structure of NETPIE2020



2 - Structure of NETPIE2020

Create a Project

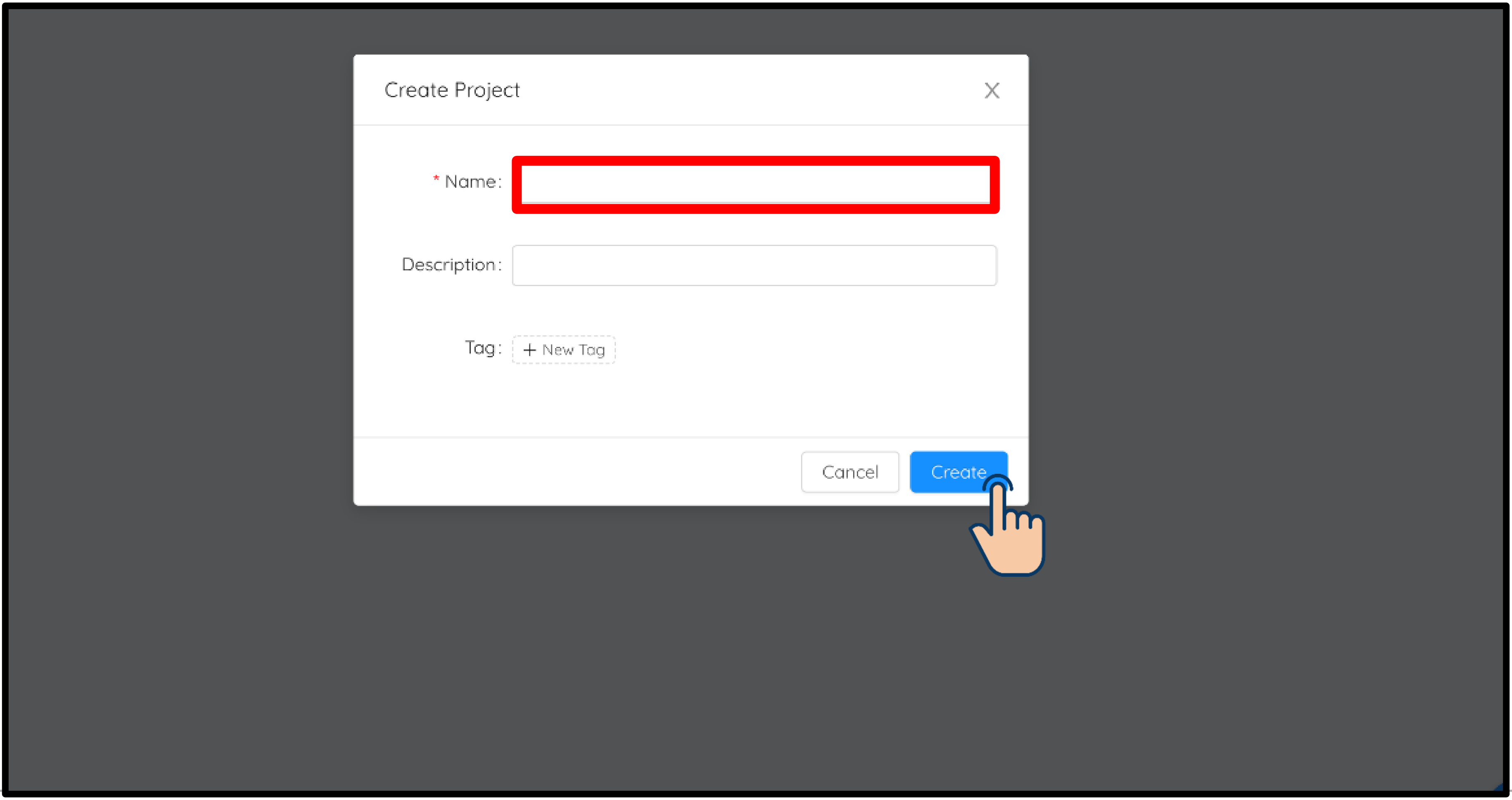
To create a project, click at [+] as shown in the picture



2 - Structure of NETPIE2020

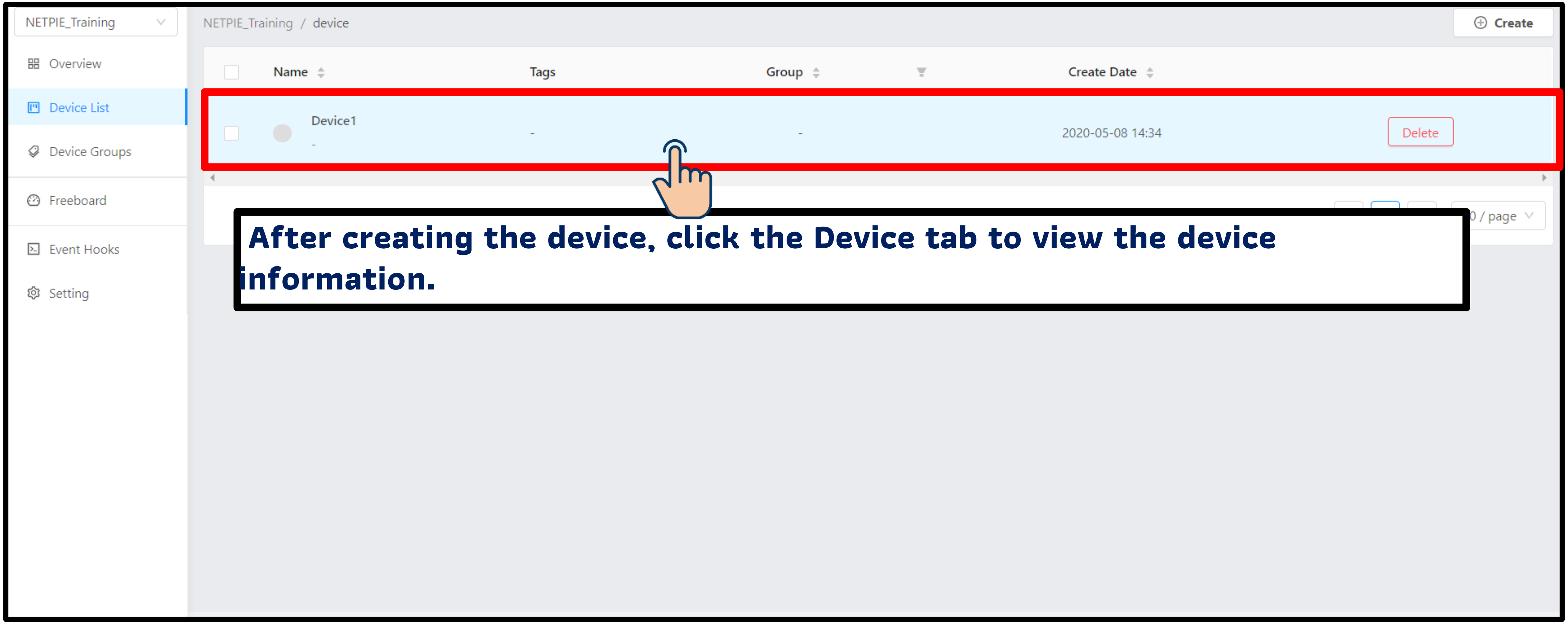
Create a Project

Fill out the information for creating the project. The filed marked with * is required.



2 - Structure of NETPIE2020

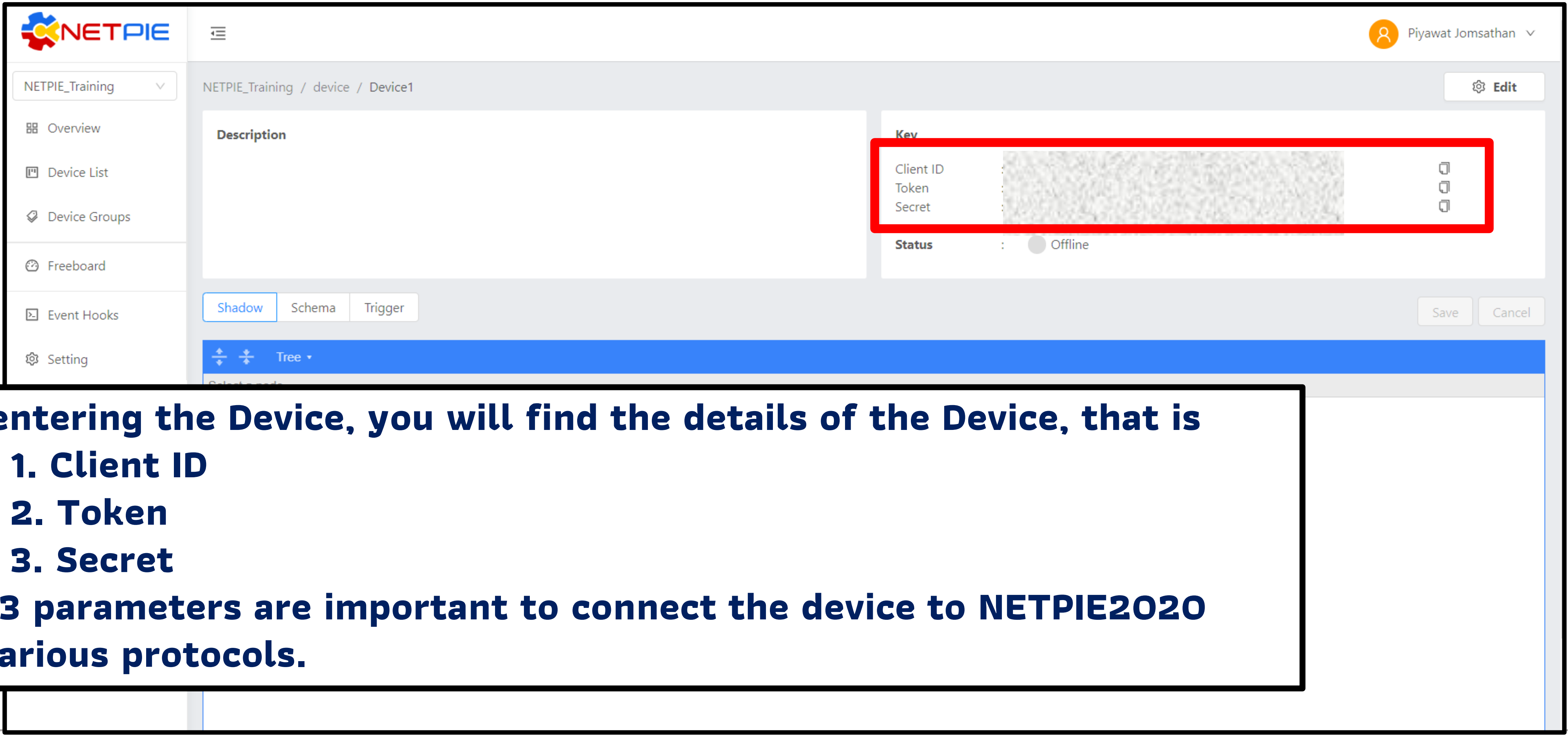
Device Creation



After creating the device, click the Device tab to view the device information.

2 - Structure of NETPIE2020

Device Creation



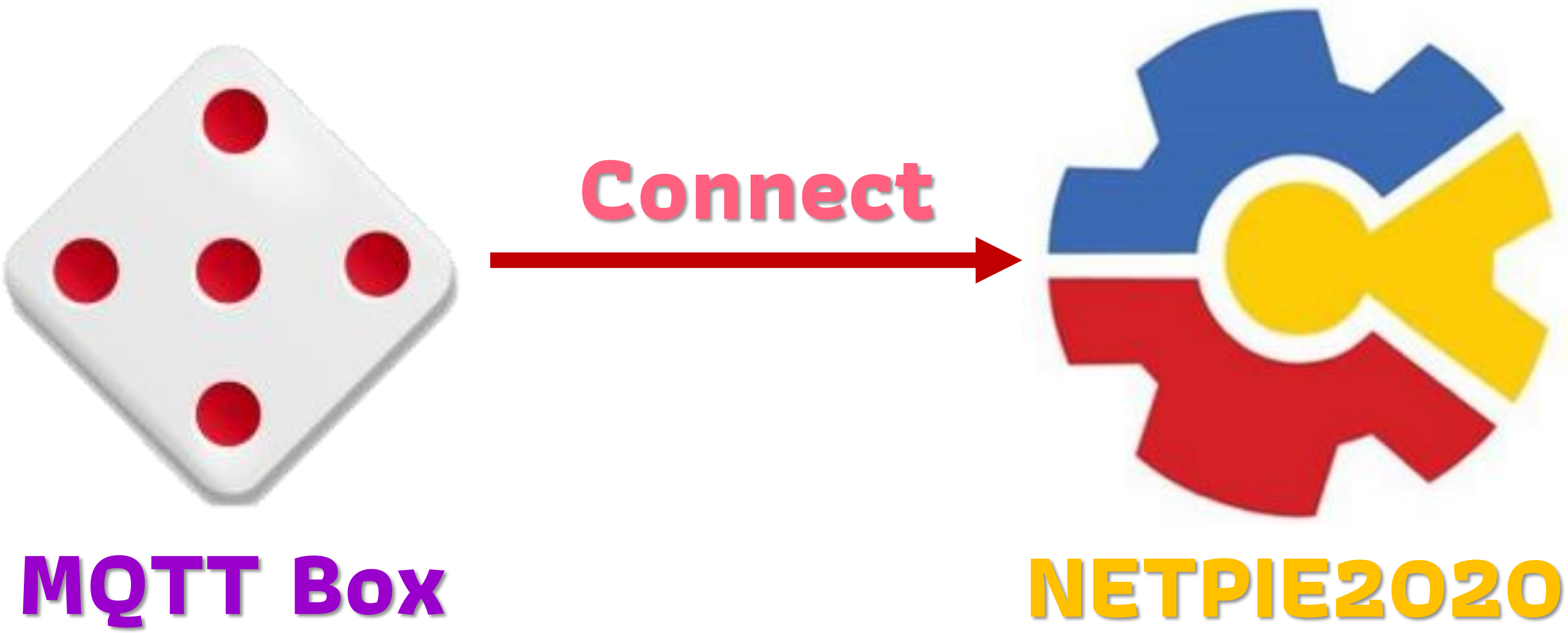
After entering the Device, you will find the details of the Device, that is

1. **Client ID**
2. **Token**
3. **Secret**

These 3 parameters are important to connect the device to NETPIE2020 using various protocols.

2 - Structure of NETPIE2020

Exercise 1 : Connecting the NETPIE2020 with MQTT Box

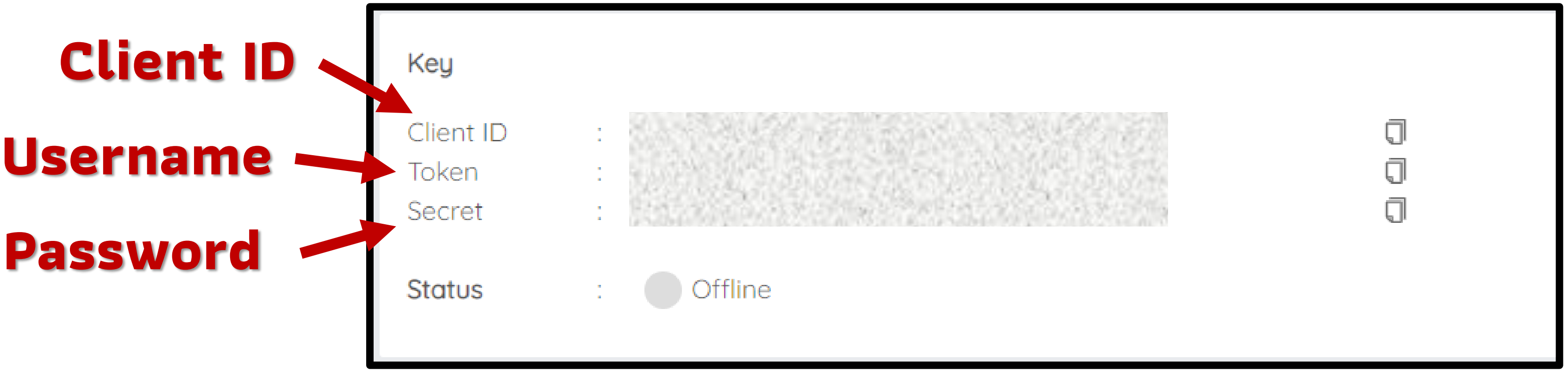


2 - Structure of NETPIE2020

Exercise 1 : Connecting the NETPIE2020 with MQTT Box

NETPIE2020 connection requires 4 parameters as follows:

1. **Host : broker.netpie.io**
2. **Client ID : Client ID of Device created in NETPIE2020 portal.**
3. **Username : Token of Device created in NETPIE2020 portal.**
4. **Password : Secret of Device created n NETPIE2020 portal.**
   [used for added security]



2 - Structure of NETPIE2020

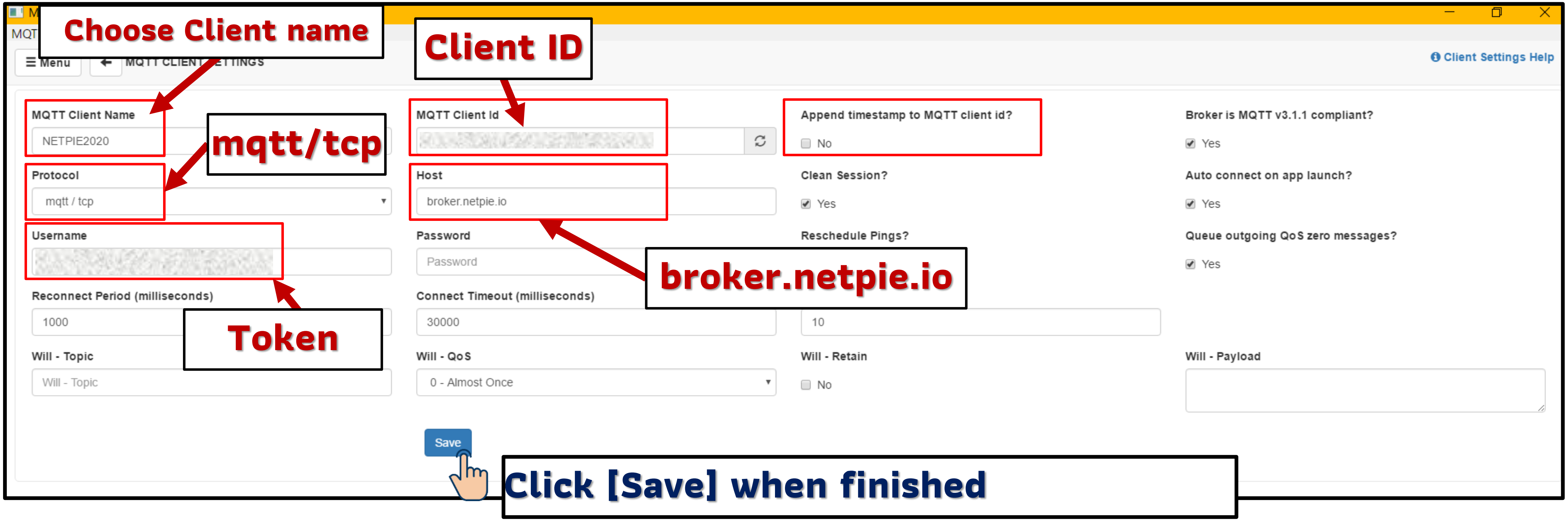
Exercise 1 : Connecting the NETPIE2020 with MQTT Box

1. Launch MQTTBox

2. Click on [Create MQTT Client] button

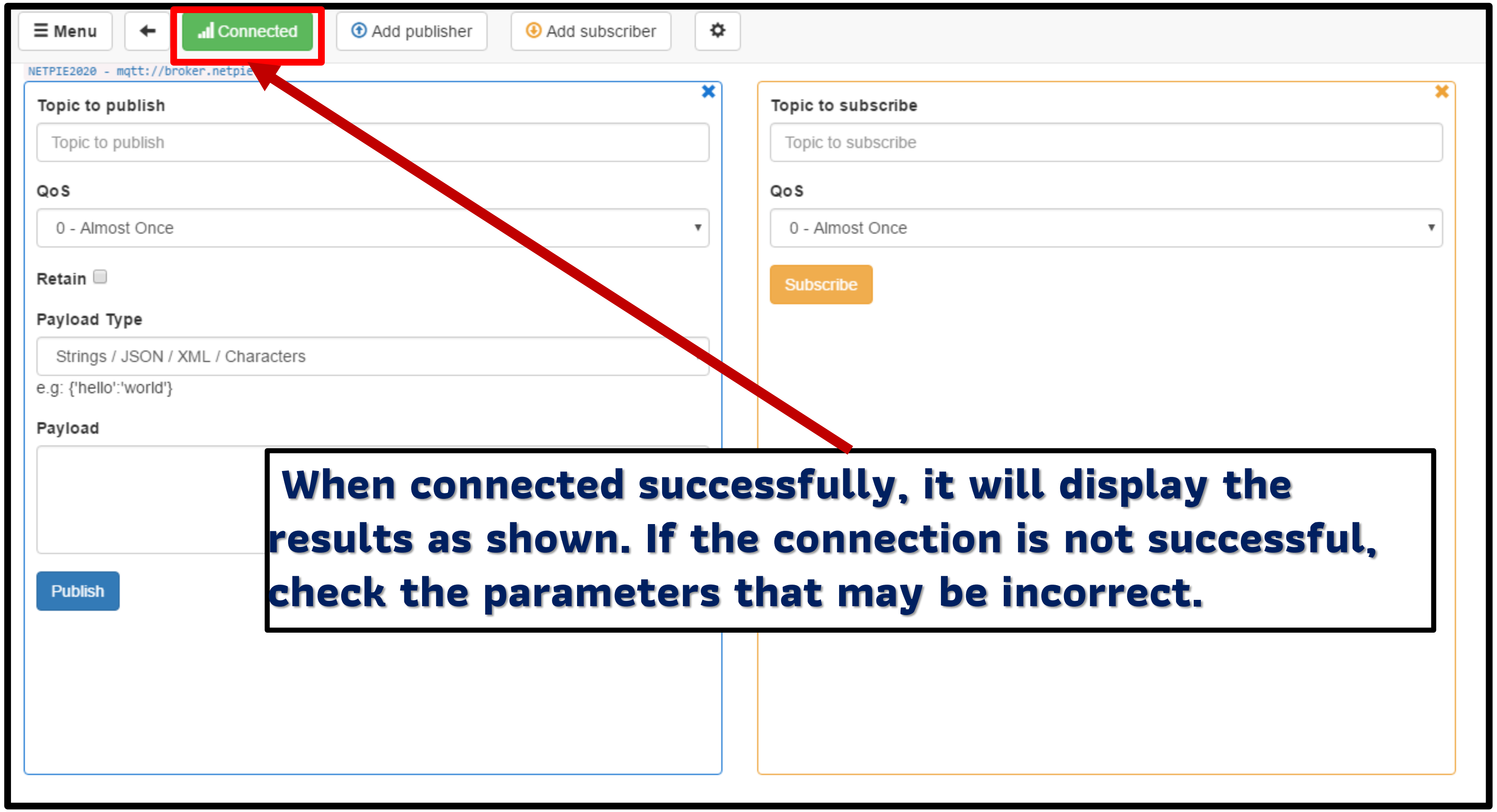
2 - Structure of NETPIE2020

Exercise 1 : Connecting the NETPIE2020 with MQTT Box



2 - Structure of NETPIE2020

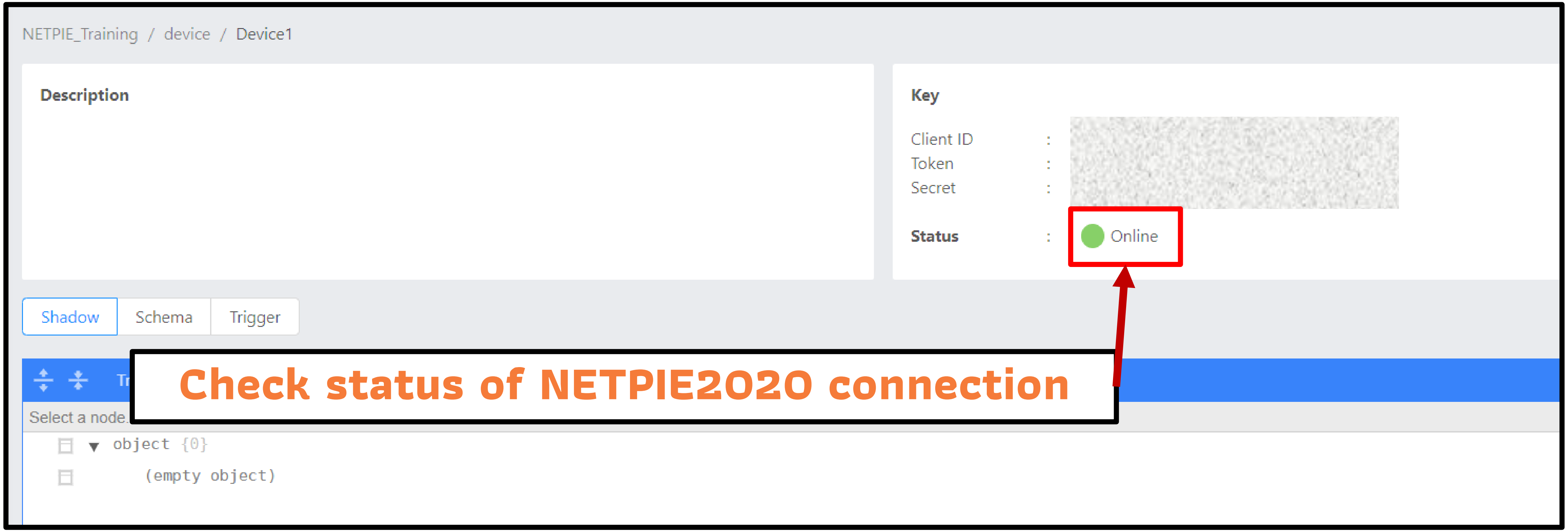
Exercise 1 : Connecting the NETPIE2020 with MQTT Box



When connected successfully, it will display the results as shown. If the connection is not successful, check the parameters that may be incorrect.

2 - Structure of NETPIE2020

Exercise 1 : Connecting the NETPIE2020 with MQTT Box



3

Type of Communication in NETPIE2020

NETPIE

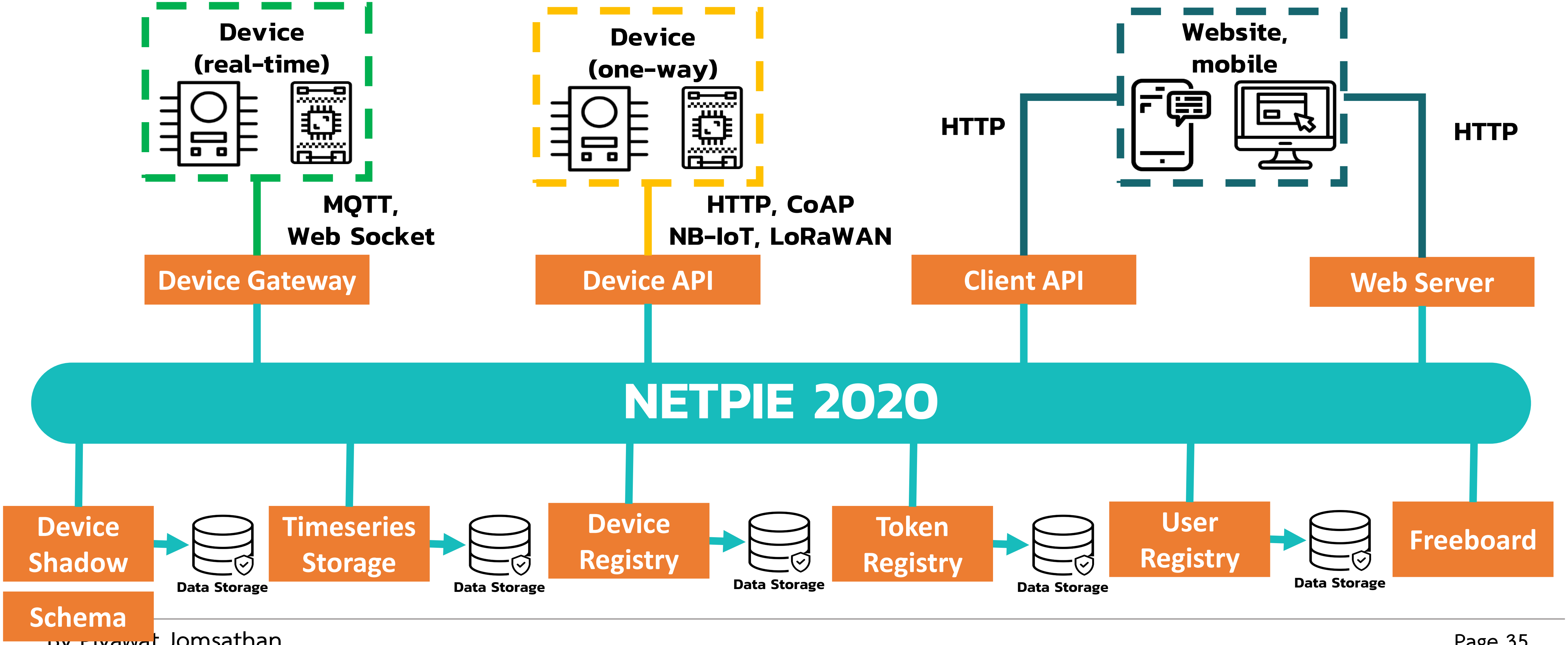
where things chat

สวทช NSTDA 30th Anniversary

NECTEC a member of NSTDA

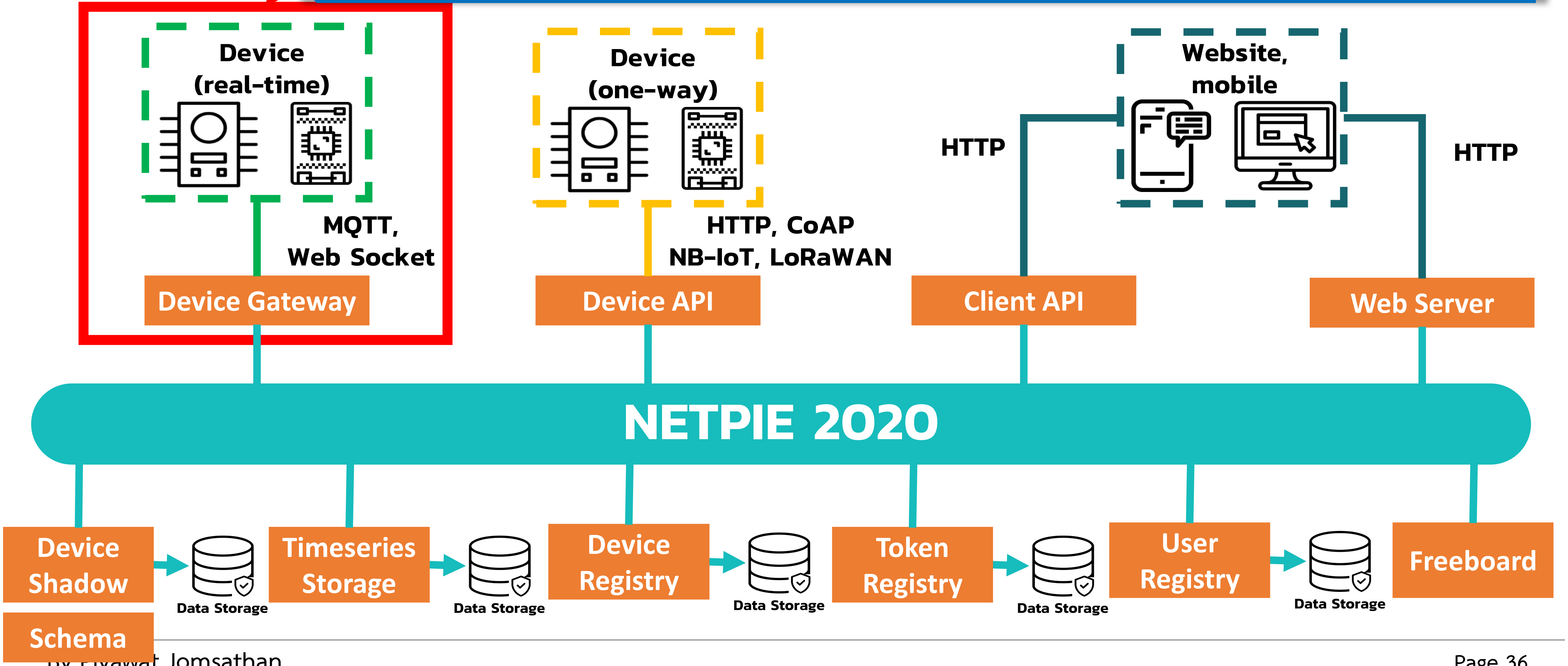
3 - Type of Communication in NETPIE2020

NETPIE 2020 Architecture



3 - Type of Communication in NETPIE2020

If real-time monitoring is required on NETPIE2020, a device must communicate only through Device Gateway with the MQTT protocol.

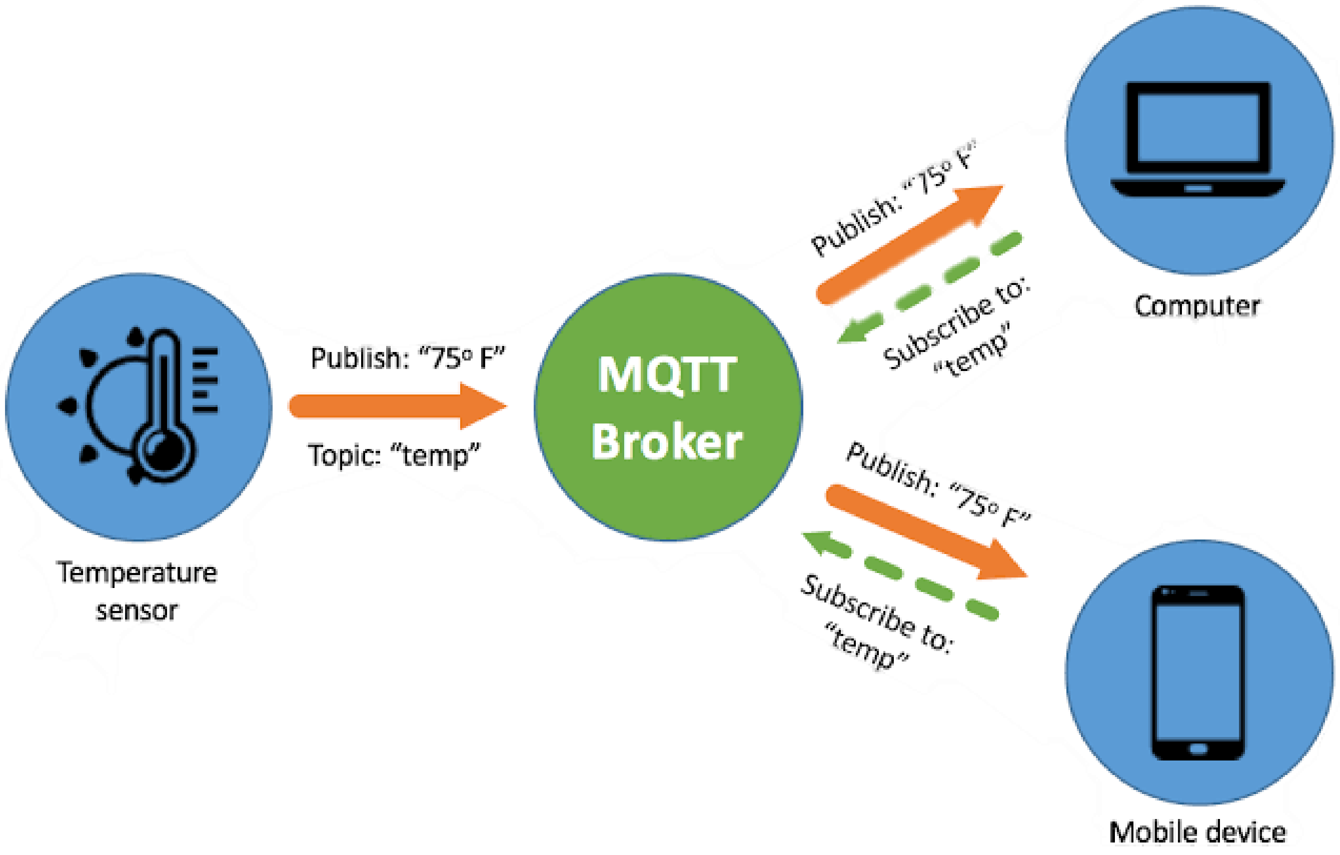


3 - Type of Communication in NETPIE2020

NETPIE2020 communication (MQTT)

What is MQTT ??

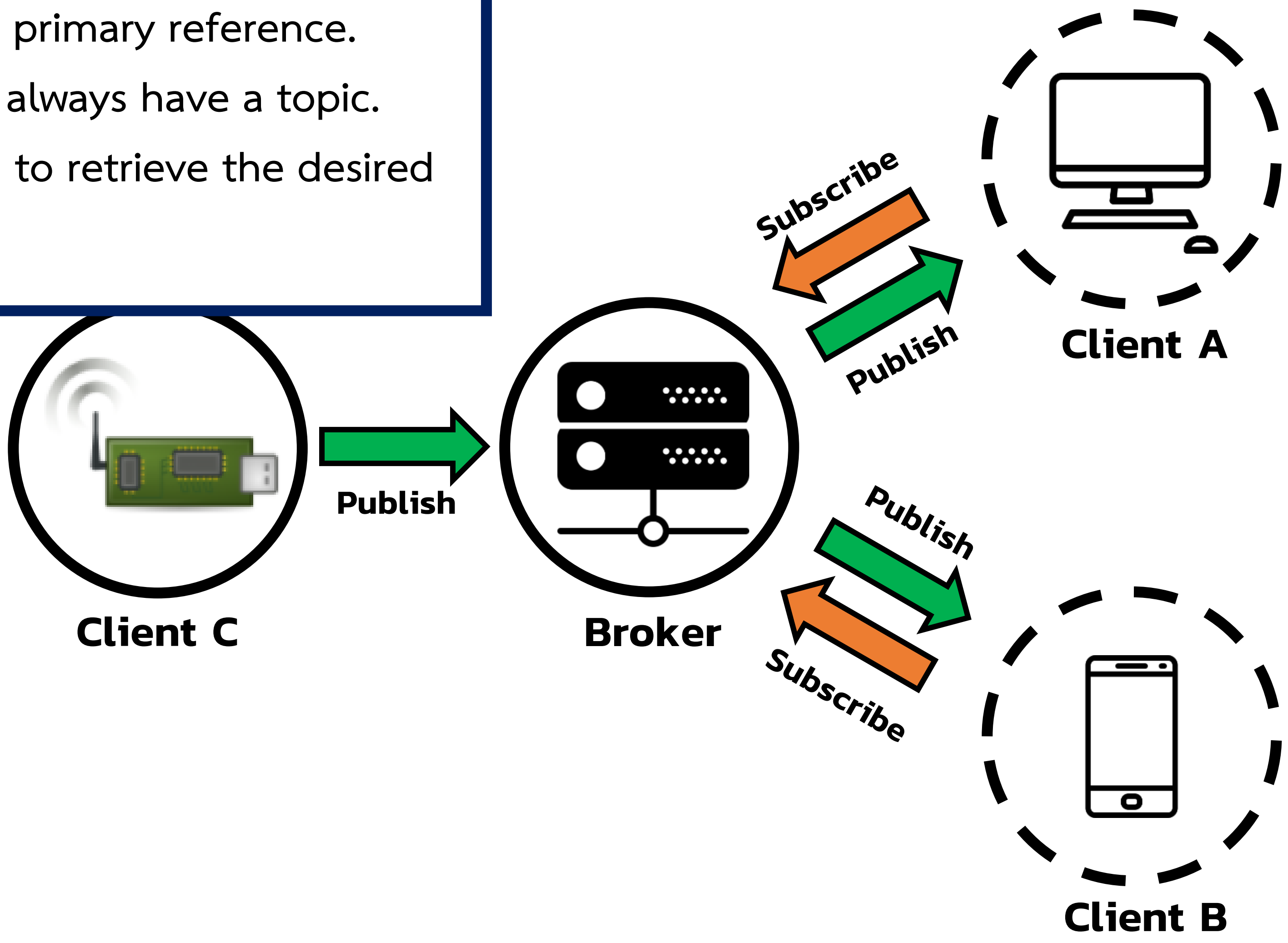
- MQTT is a protocol designed to connect M2M or device to device, which is supported by IoT
- Use the principle of data transmission, Publish / Subscribe, similar to the principles used in the Web Service that requires a Web Server as an intermediary between users' computers.
- But MQTT uses an intermediary called Broker that manages the transmission order between devices and Publish / Subscribe mechanisms.



3 - Type of Communication in NETPIE2020

transmissions on the MQTT have a topic as their primary reference.

- The data to be published to the Broker must always have a topic.
- On the subscribe end, it must refer to a topic to retrieve the desired information.



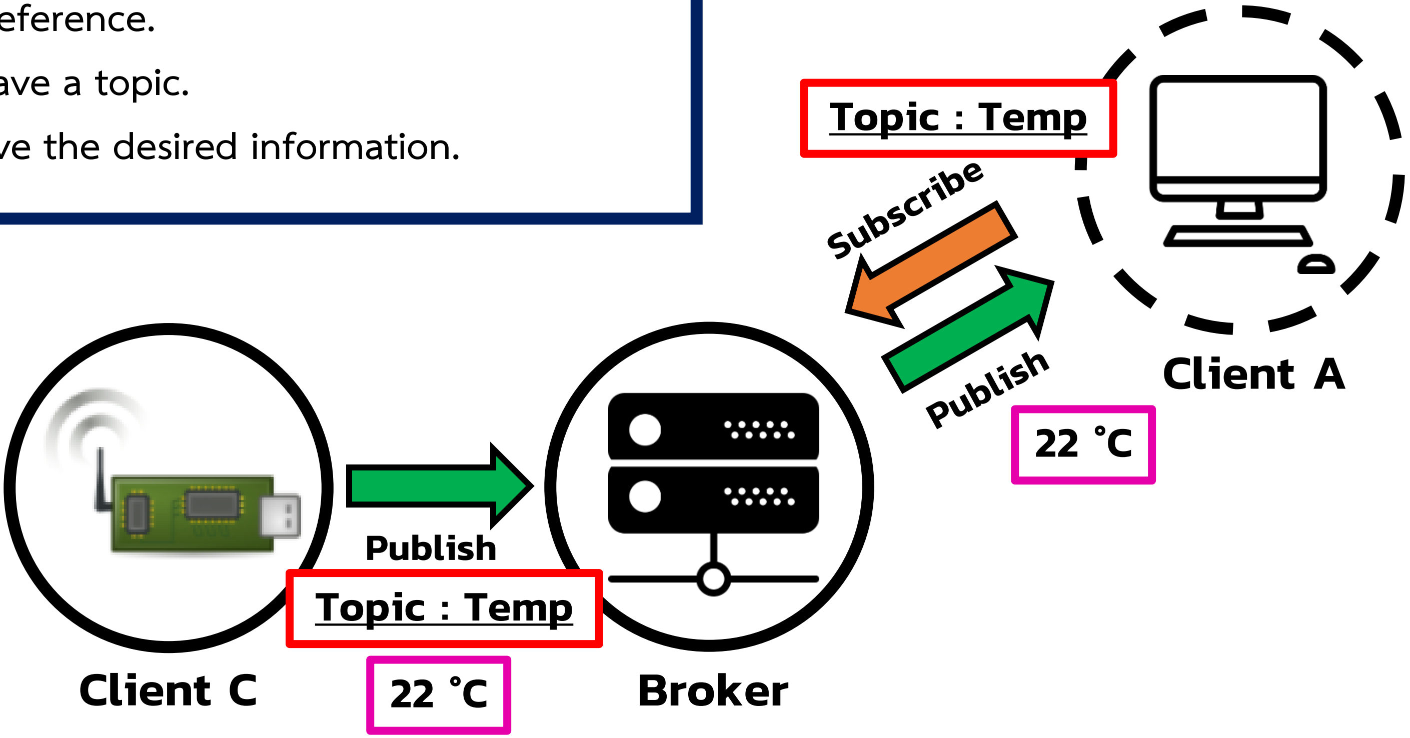
3 - Type of Communication in NETPIE2020

transmissions on the MQTT have a topic as their primary reference.

- The data to be published to the Broker must always have a topic.
- On the subscribe end, it must refer to a topic to retrieve the desired information.

MQTT communication example

- Client C publishes information on a topic named Temp, while Client A subscribes to topic Temp.
- The data from Client C is a temperature with values equal to 22 ° C, so Client A receives the information.



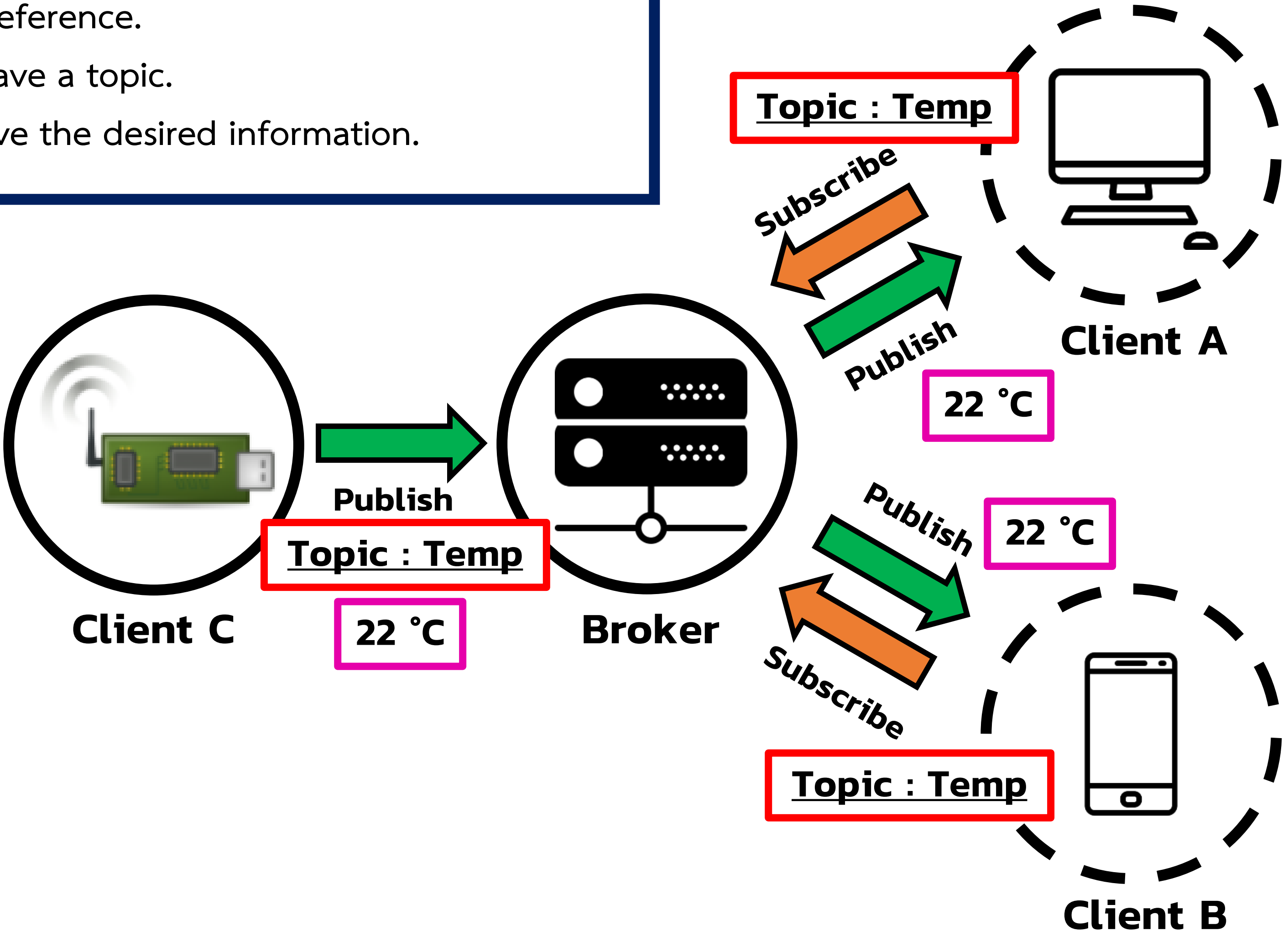
3 - Type of Communication in NETPIE2020

transmissions on the MQTT have a topic as their primary reference.

- The data to be published to the Broker must always have a topic.
- On the subscribe end, it must refer to a topic to retrieve the desired information.

MQTT communication example

- Client C publishes information on a topic named Temp, while Client A subscribes to topic Temp.
- The data from Client C is a temperature with values equal to 22 ° C, so Client A receives the information.
- Later, Client B subscribes to topic Temp as well. The stored temperature is immediately sent to Client B after subscribing.



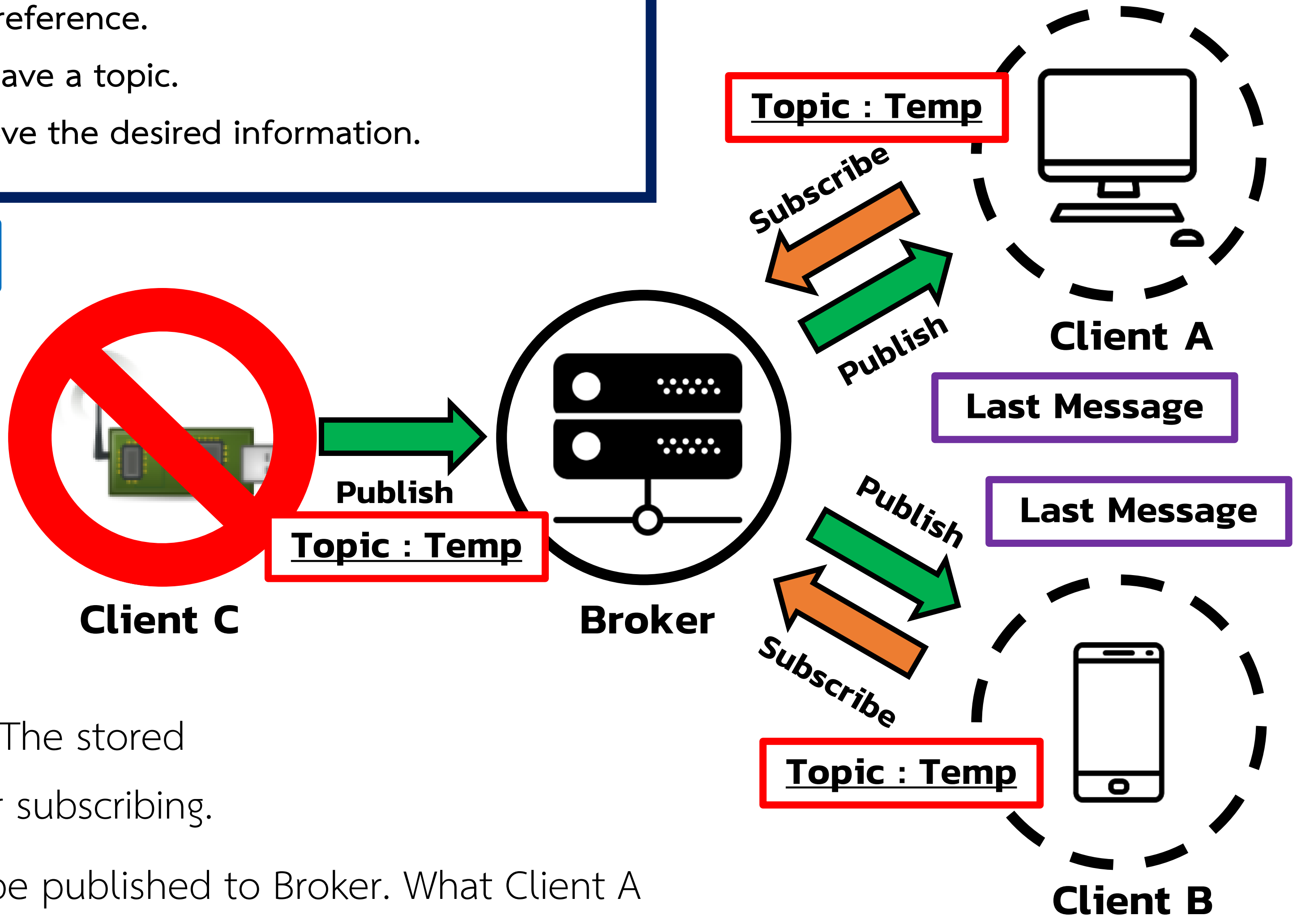
3 - Type of Communication in NETPIE2020

transmissions on the MQTT have a topic as their primary reference.

- The data to be published to the Broker must always have a topic.
- On the subscribe end, it must refer to a topic to retrieve the desired information.

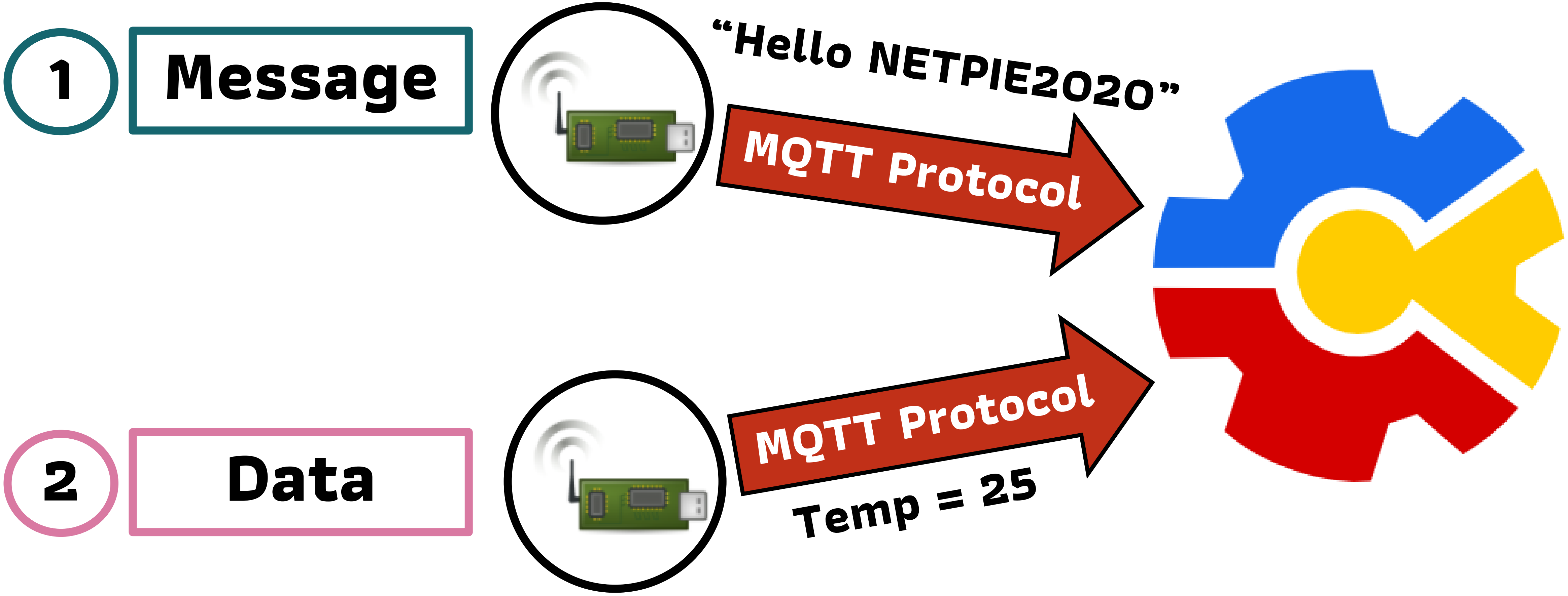
MQTT communication example

- Client C publishes information on a topic named Temp, while Client A subscribes to topic Temp.
- The data from Client C is a temperature with values equal to 22 ° C, so Client A receives the information.
- Later, Client B subscribes to topic Temp as well. The stored temperature is immediately sent to Client B after subscribing.
- But when Client C is disconnected, no data will be published to Broker. What Client A and B display is the last message sent by Client C



3 - Type of Communication in NETPIE2020

There are two types of messages sent to NETPIE2020 via MQTT.



3 - Type of Communication in NETPIE2020

MQTT on NETPIE2020 has two types of transmission

1 **Message**

Send a message via MQTT by referring to a topic.
The format is **@msg/topic**

2 **Shadow [Data]**

Send message via MQTT to Device Shadow, to record latest message. The topic is **@shadow/data/update**
with message in JSON format. For example, **{data:{temp:24}}**

3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box [Publish & Subscribe]

3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box (Publish & Subscribe)

Understand communication patterns on NETPIE2020

To send a message NETPIE2020 we need to reference a subject heading called Topic

Topic : @msg/test
“Hello NETPIE2020”

MQTTBox sends a message to NETPIE2020 with the phrase "Hello NETPIE2020”
Publish

NETPIE2020 acts like a chatroom
Broker

3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box (Publish & Subscribe)

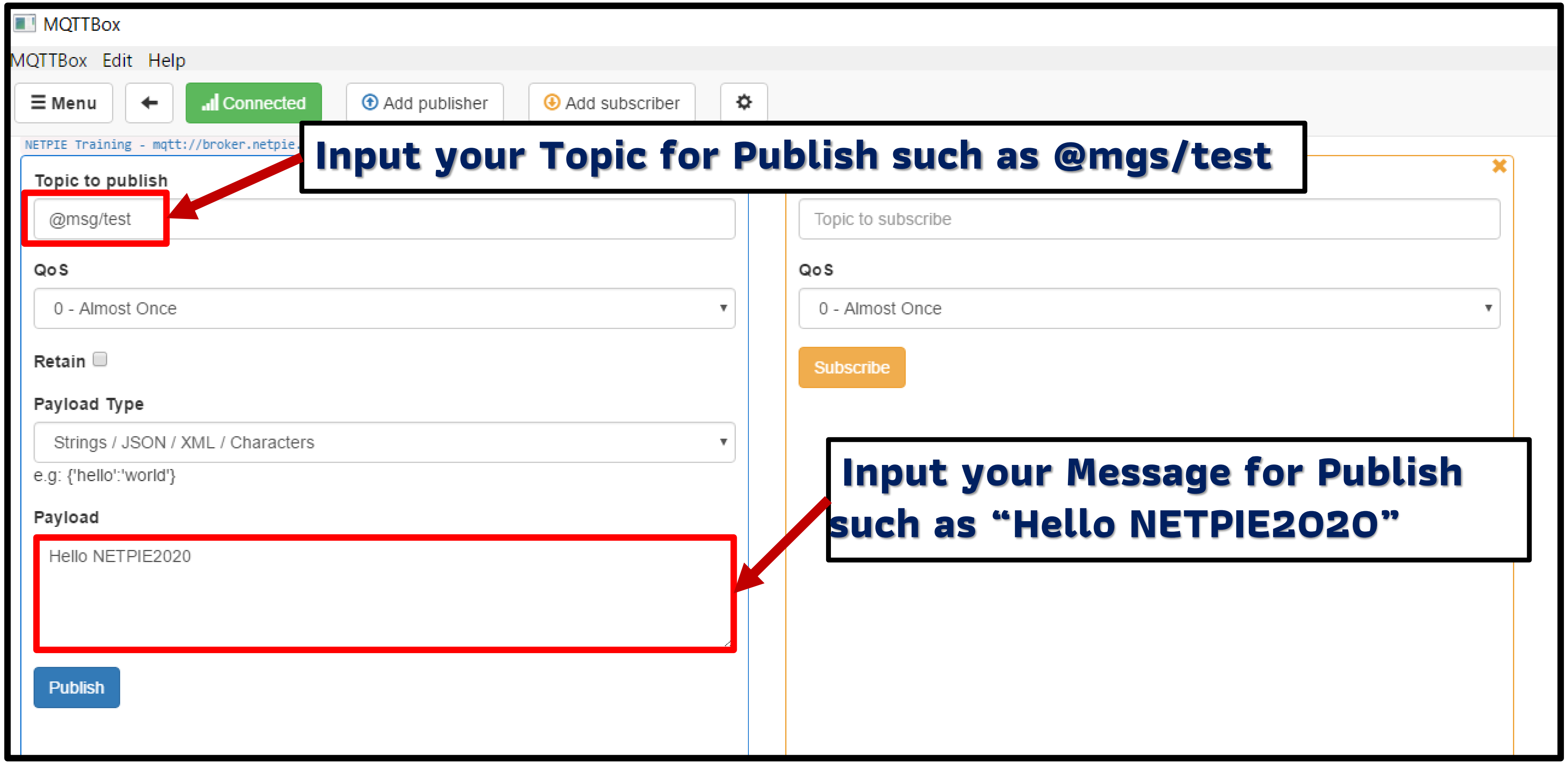
Understand communication patterns on NETPIE2020

MQTT Box Publish a message to Topic @msg/test

MQTT Box wants the message that MQTT Box sent, so must state that requirement on the Topic with a process called Subscribe

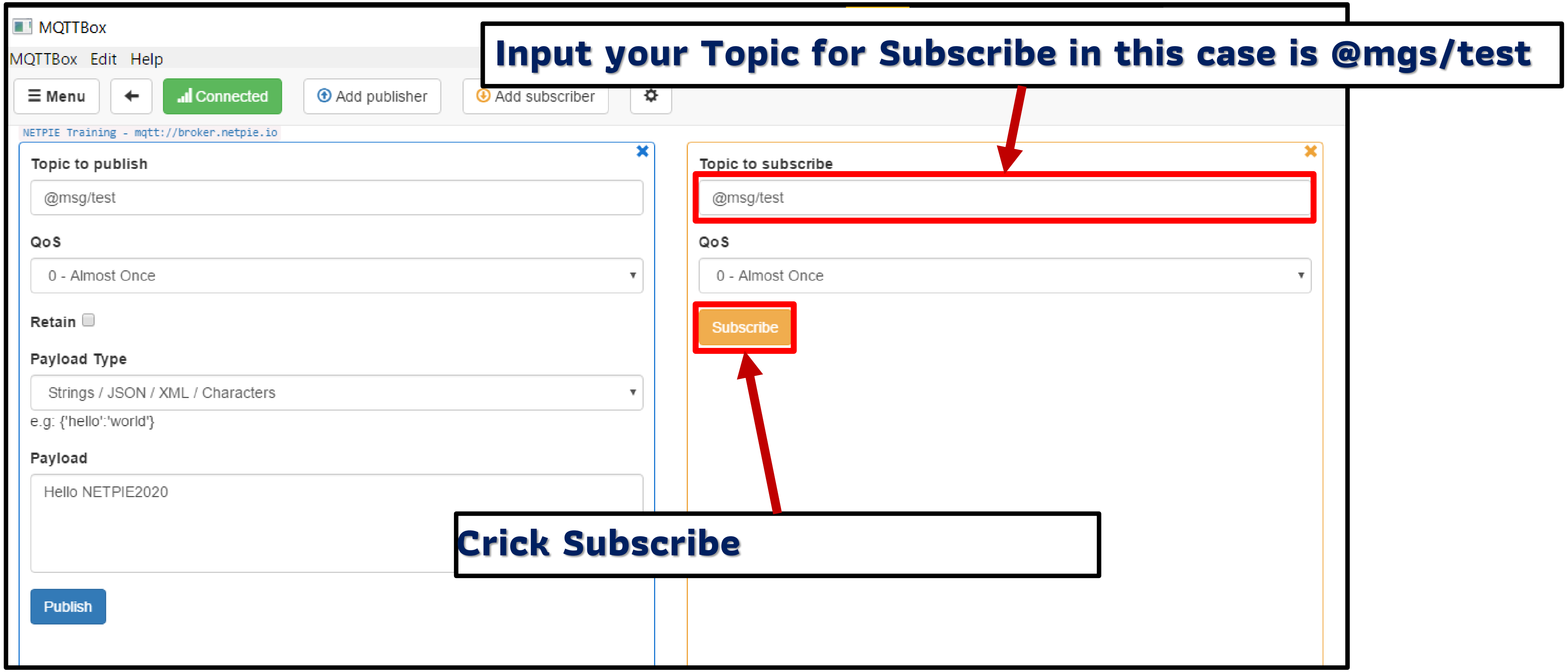
3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box (Publish & Subscribe)



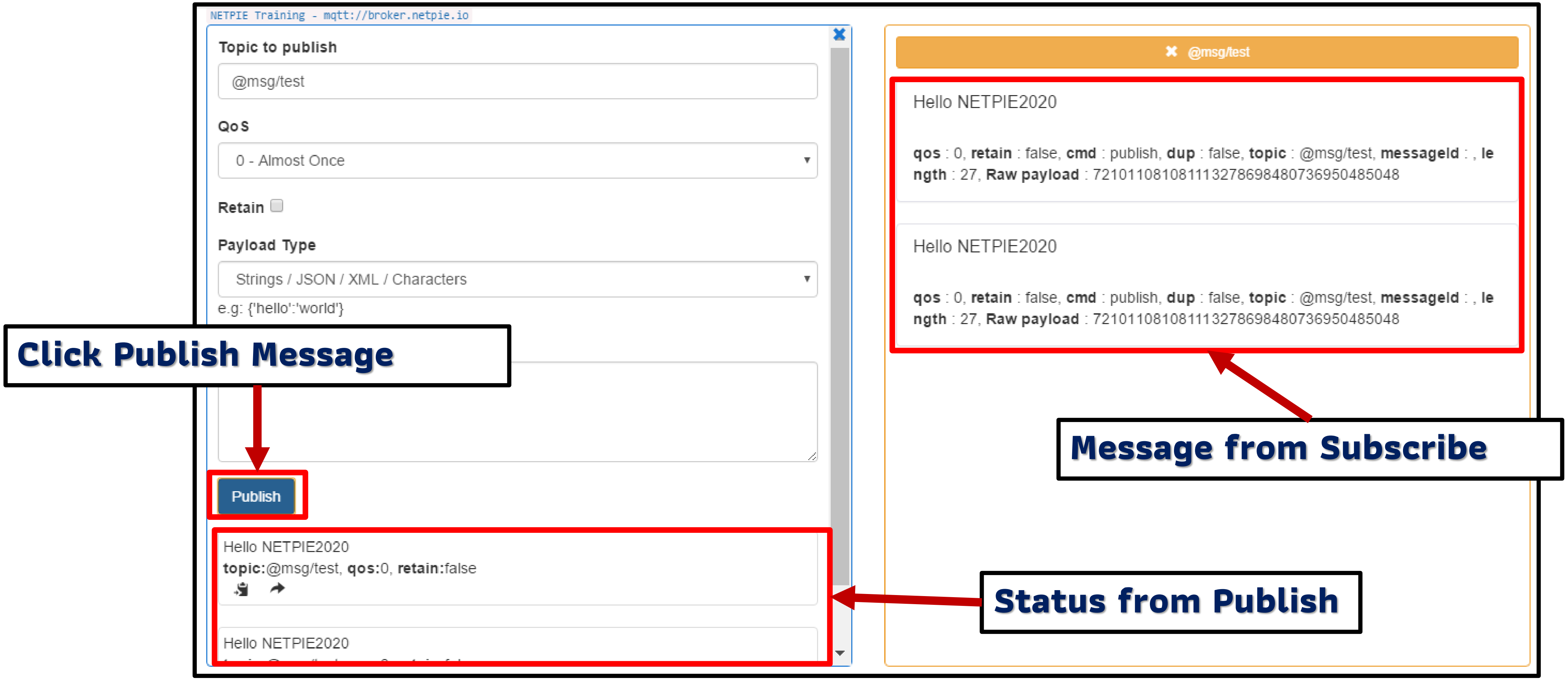
3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box [Publish & Subscribe]



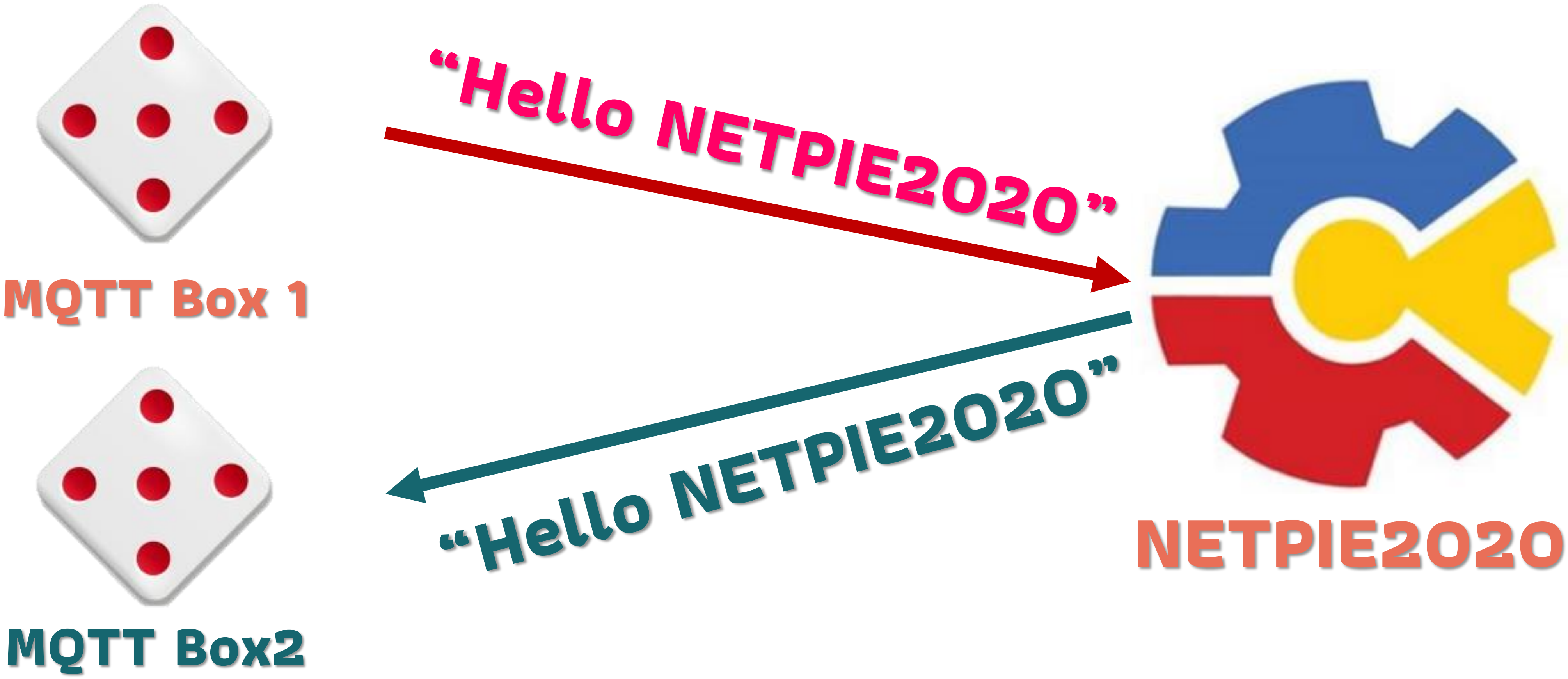
3 - Type of Communication in NETPIE2020

Exercise 2 : Communication on NETPIE2020 with MQTT Box [Publish & Subscribe]



3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

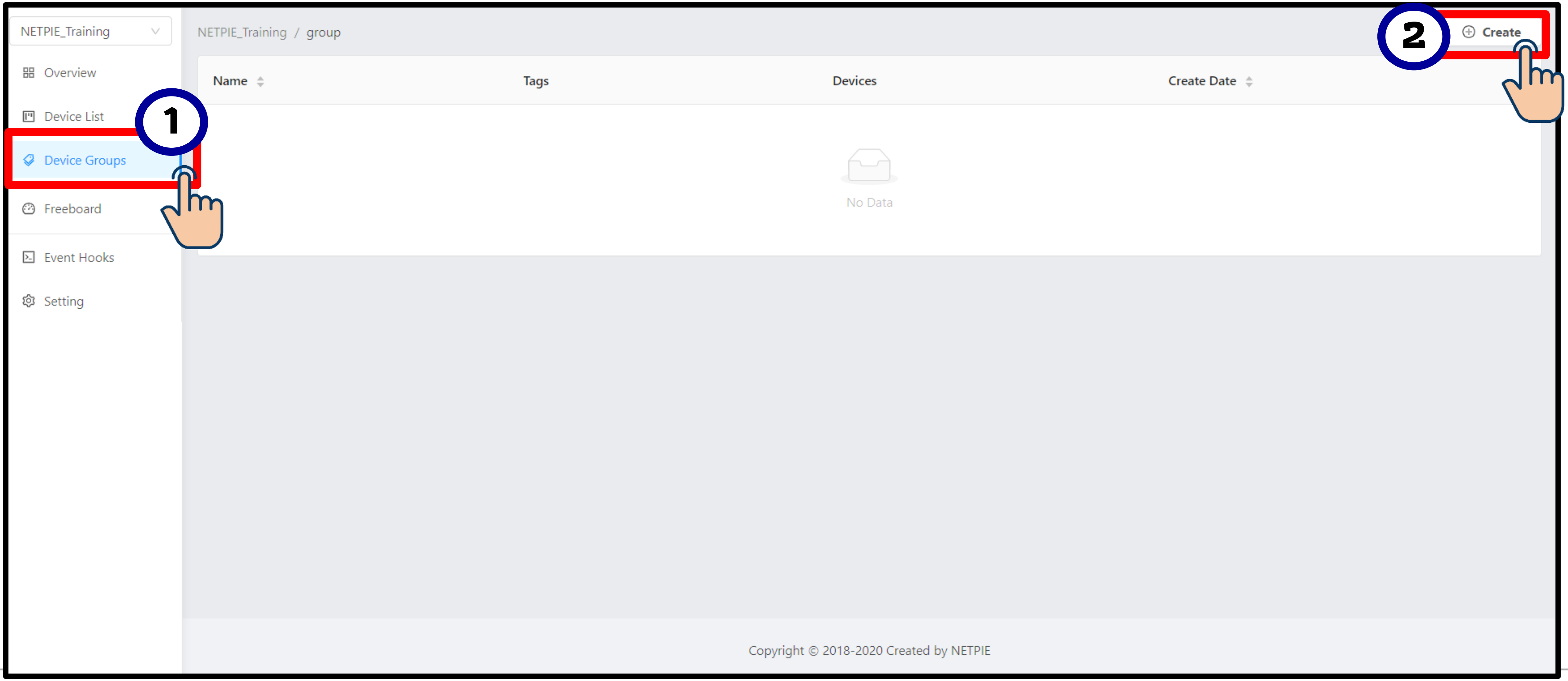
Exercise 3 : Communication on NETPIE2020 with Device Group

Understand communication patterns on NETPIE2020

The two devices must be on the same group

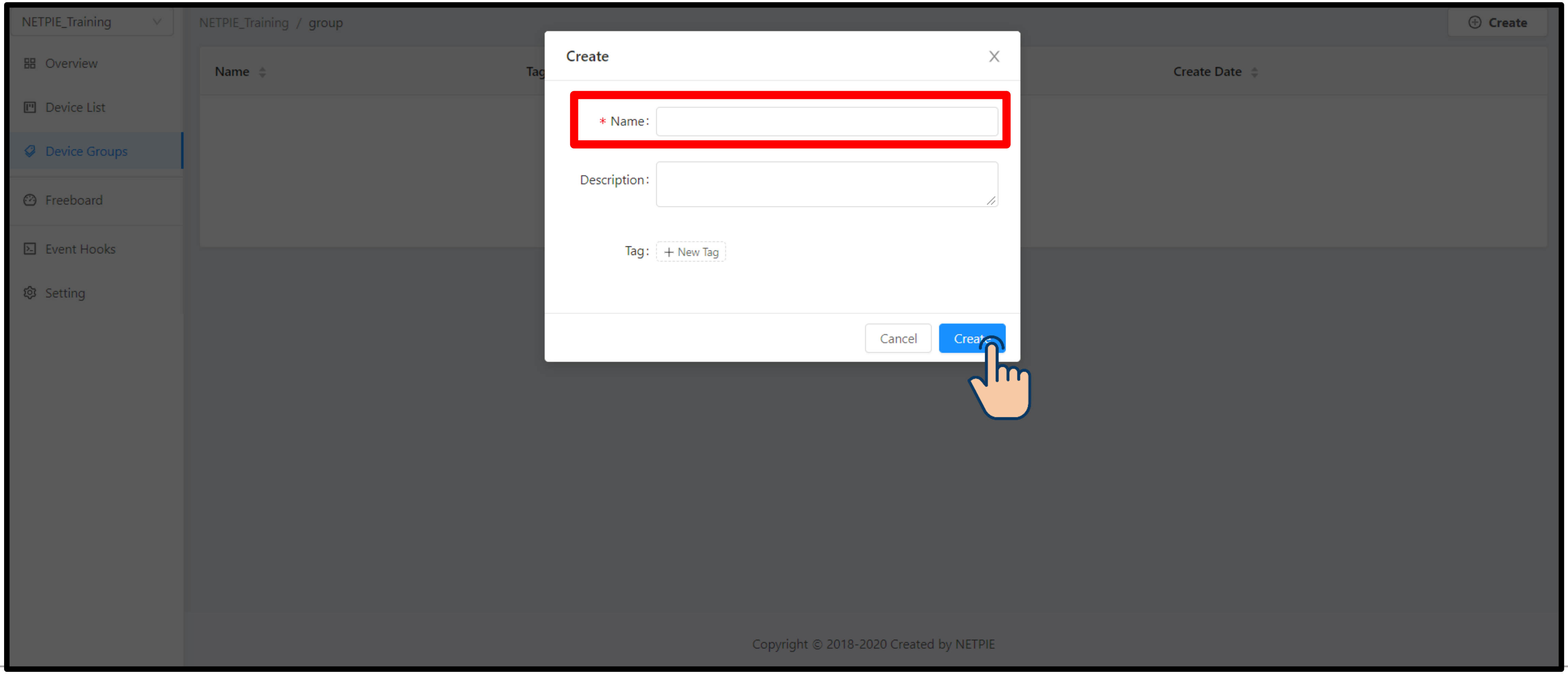
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



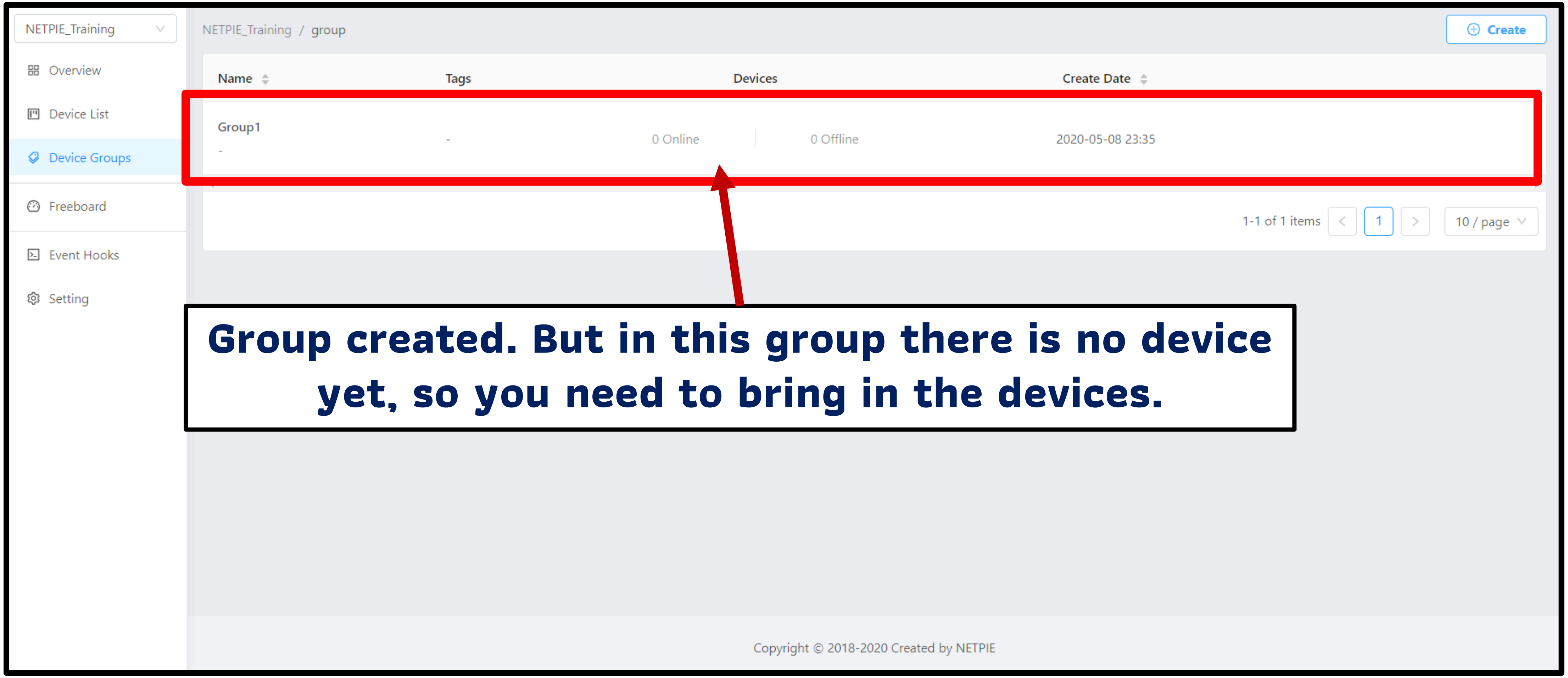
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



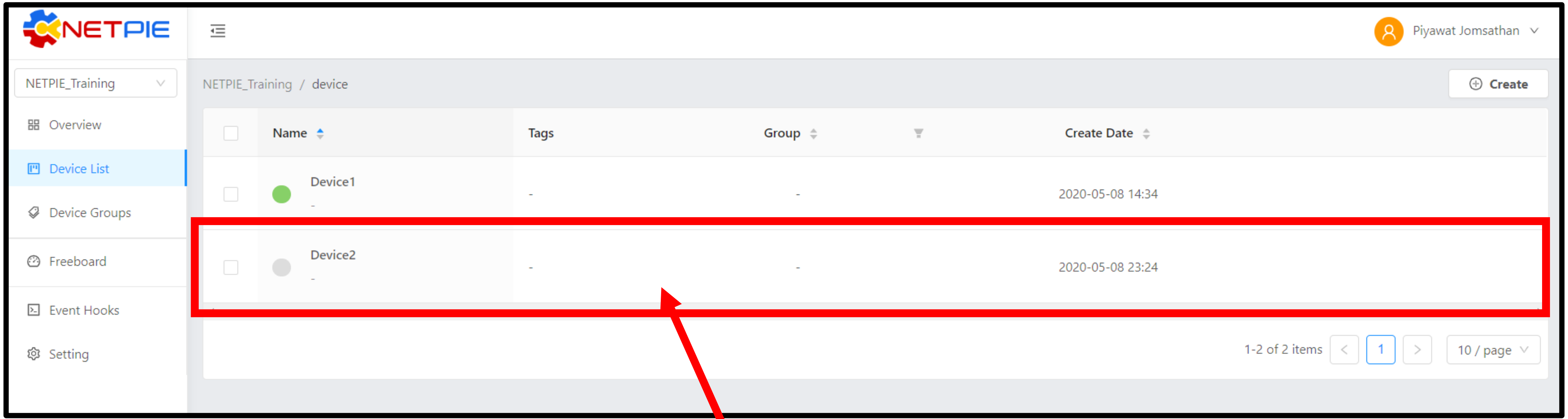
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

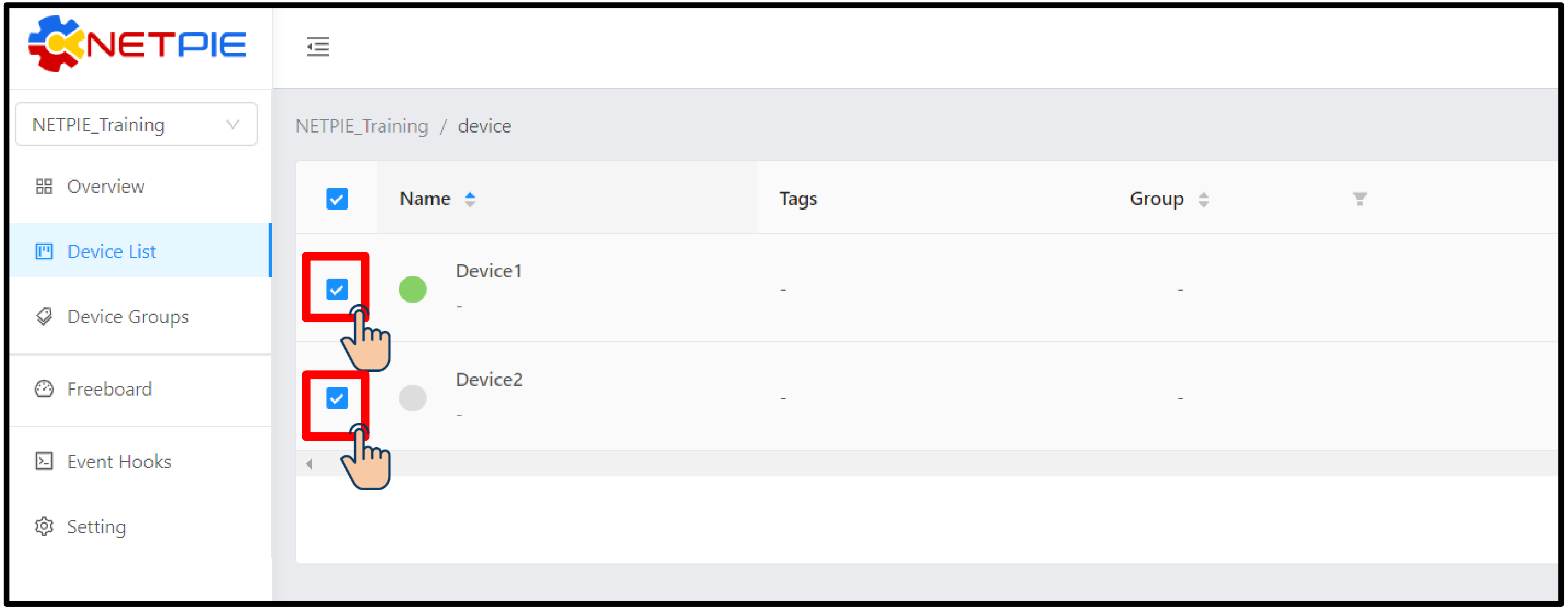
Exercise 3 : Communication on NETPIE2020 with Device Group



Create a new device for MQTTBox

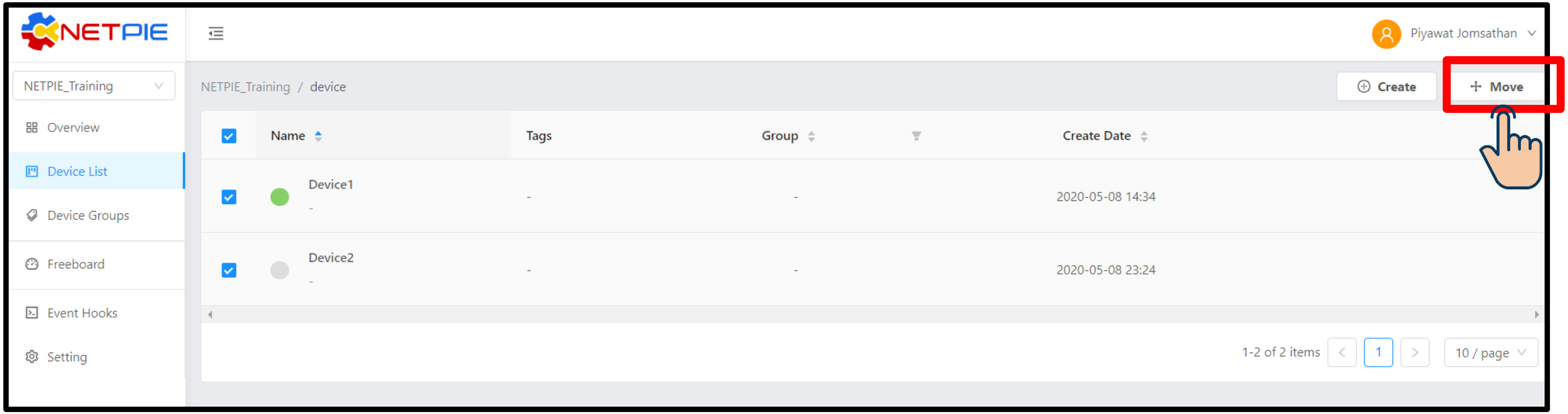
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



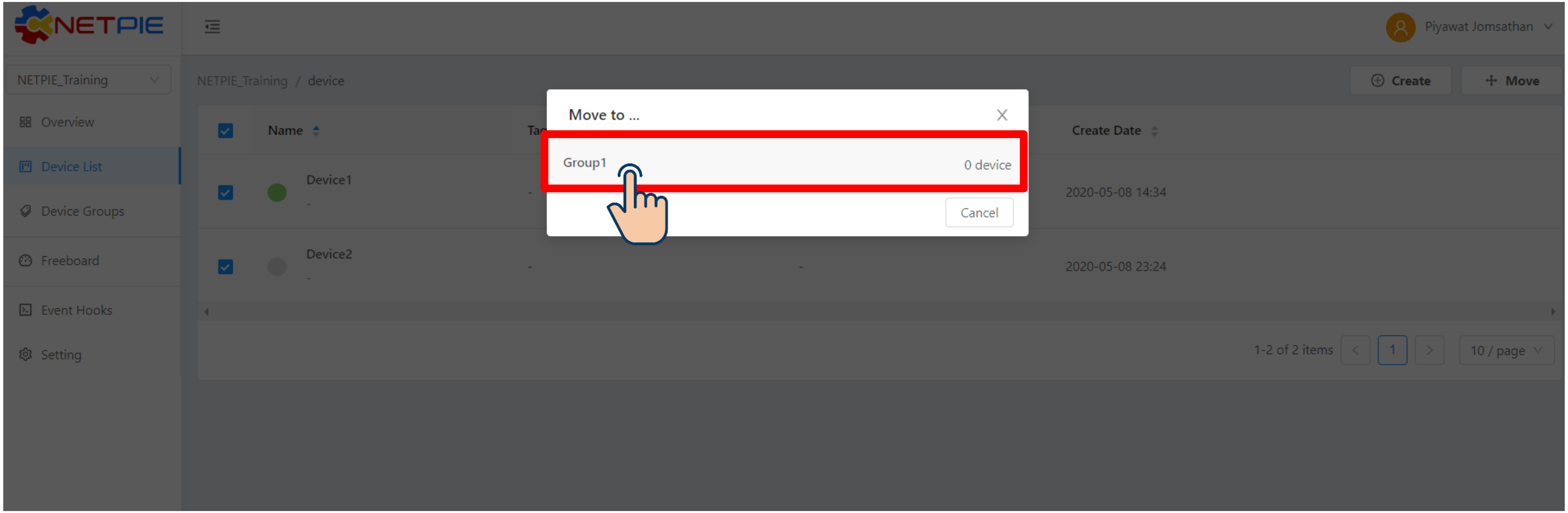
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



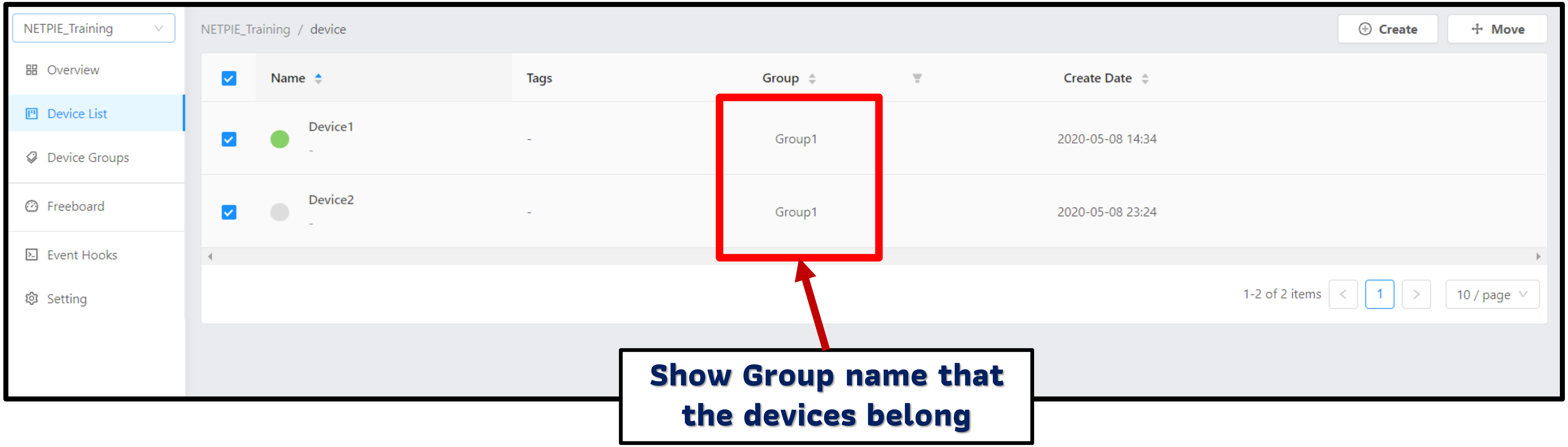
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

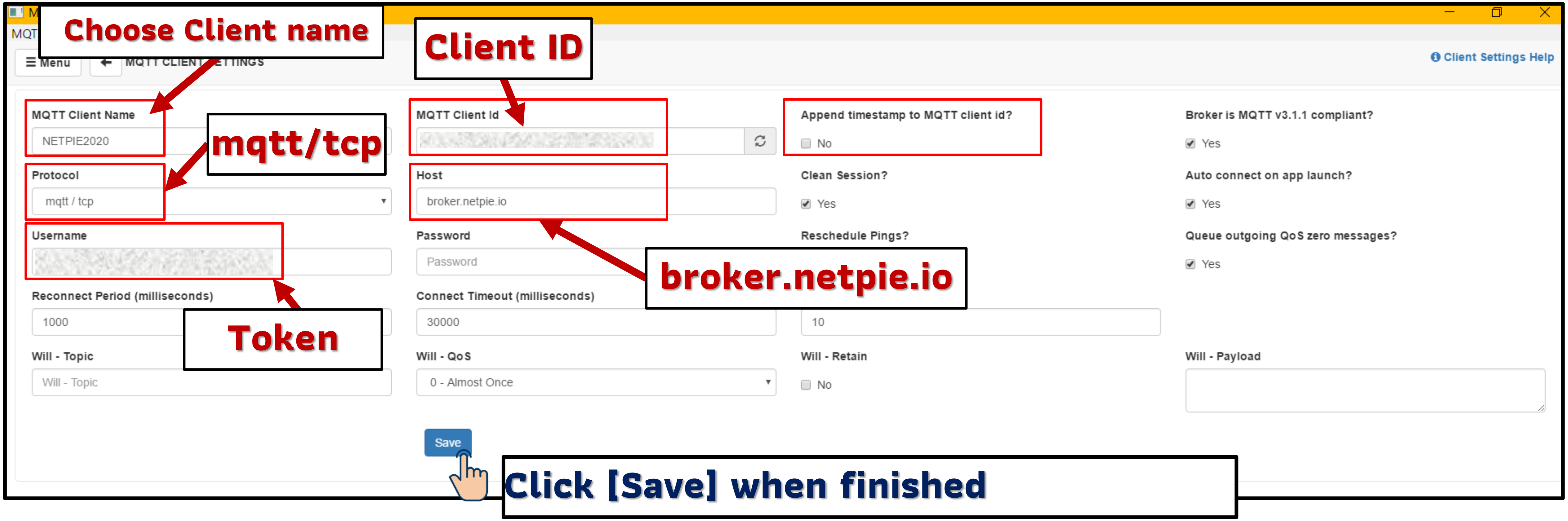
Exercise 3 : Communication on NETPIE2020 with Device Group

1. Launch new MQTTBox

2. Click on [Create MQTT Client] button

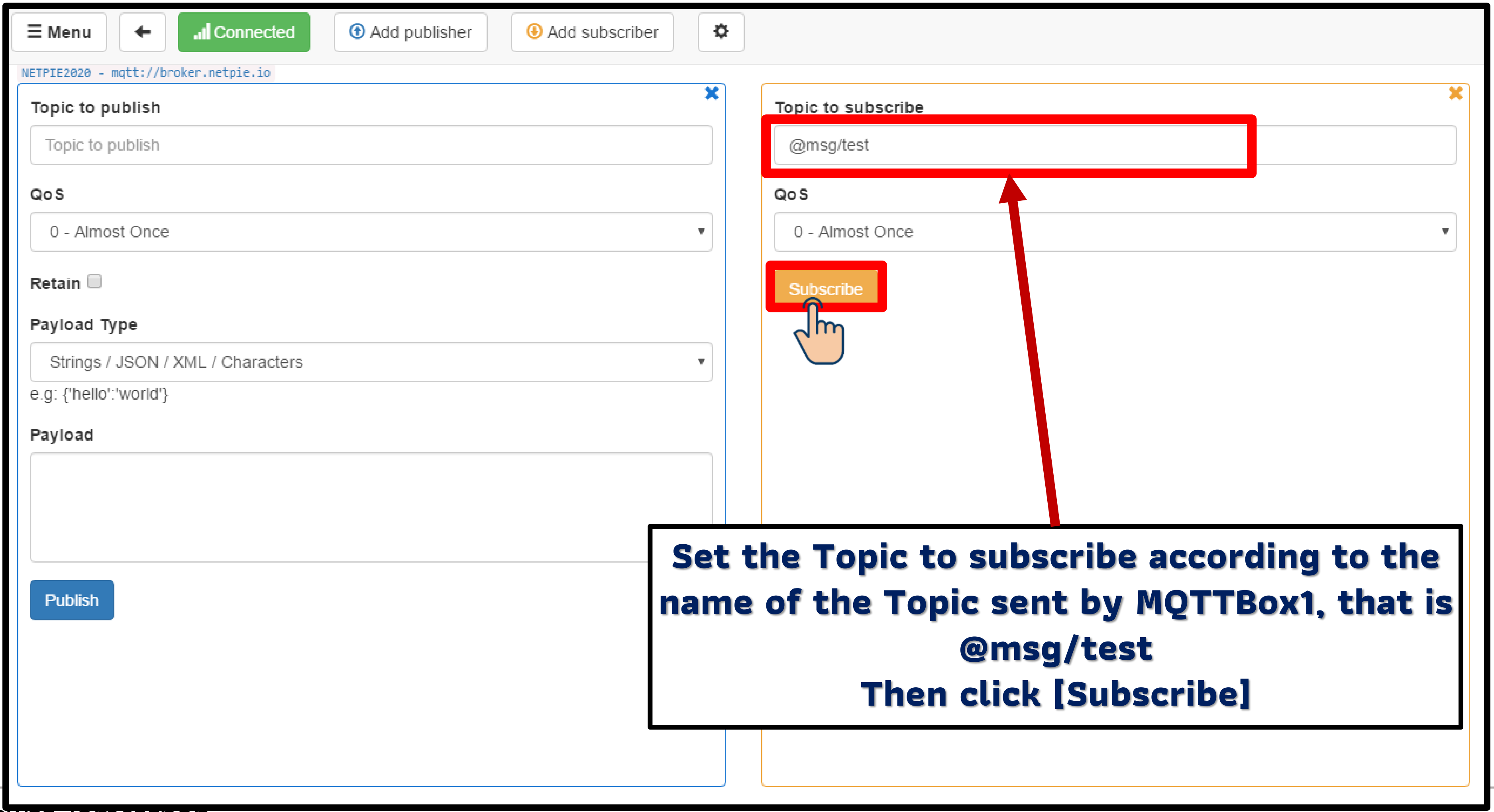
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

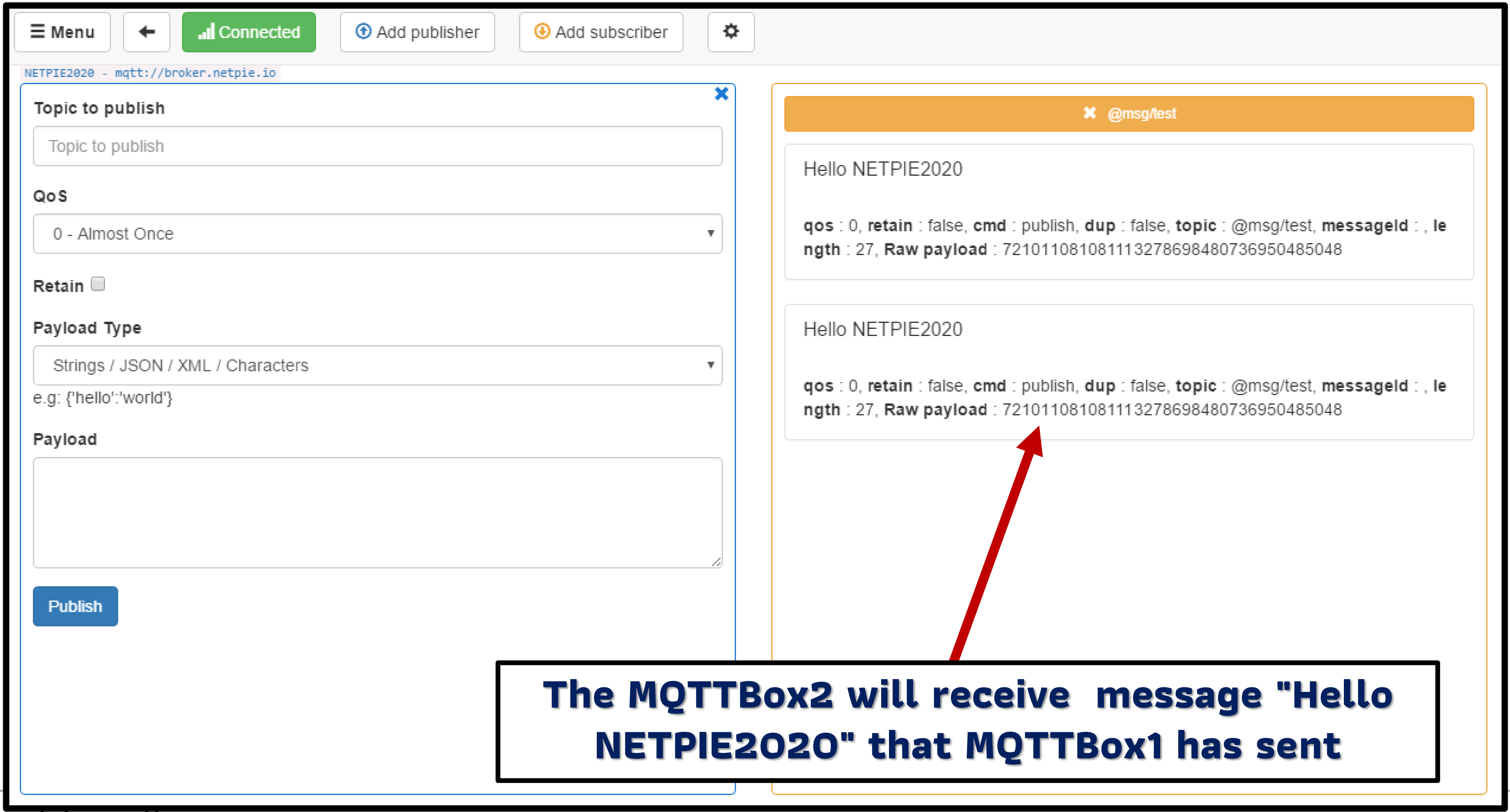
Exercise 3 : Communication on NETPIE2020 with Device Group



Set the Topic to subscribe according to the name of the Topic sent by MQTTBox1, that is @msg/test
Then click [Subscribe]

3 - Type of Communication in NETPIE2020

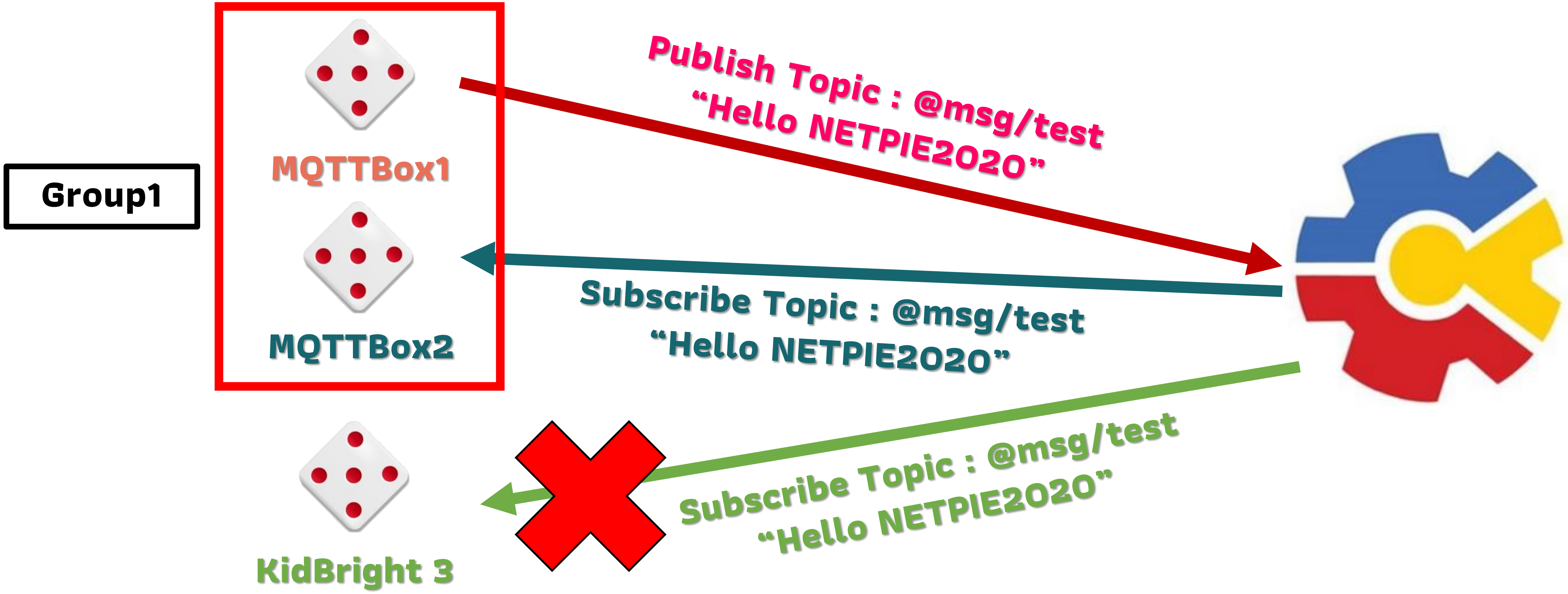
Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

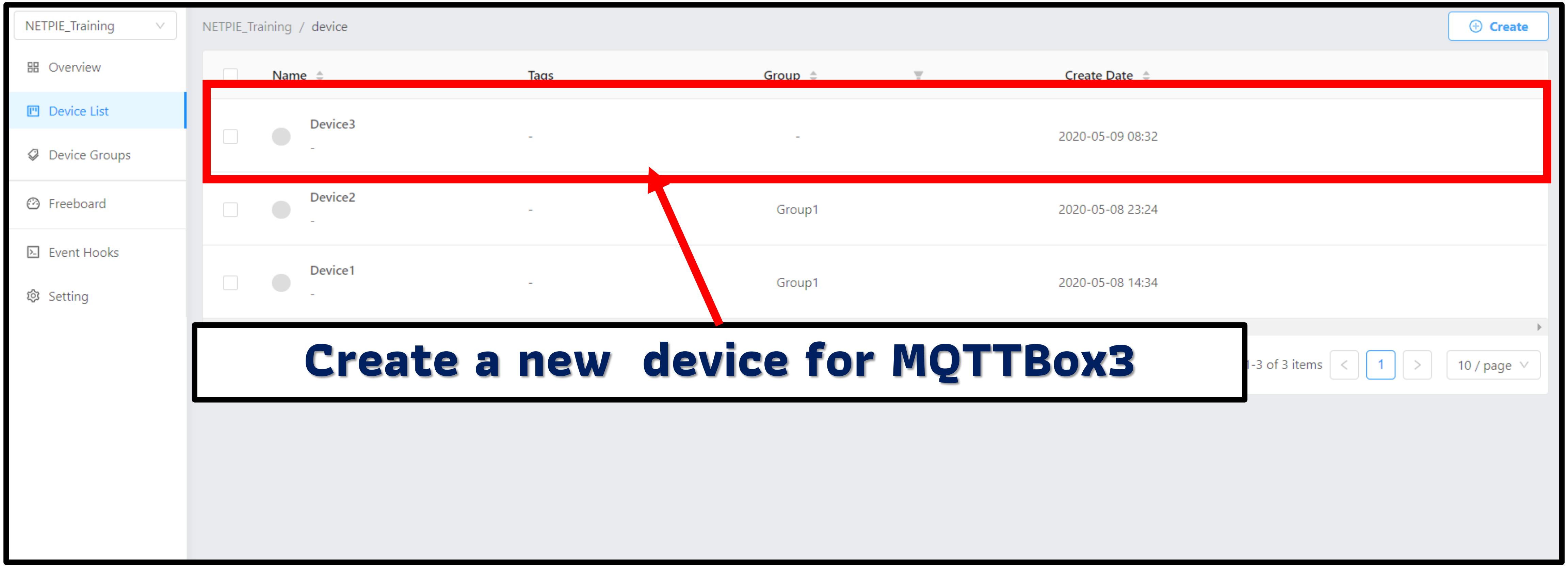
Exercise 3 : Communication on NETPIE2020 with Device Group

In case a device not belong to the group subscribes to this topic



3 - Type of Communication in NETPIE2020

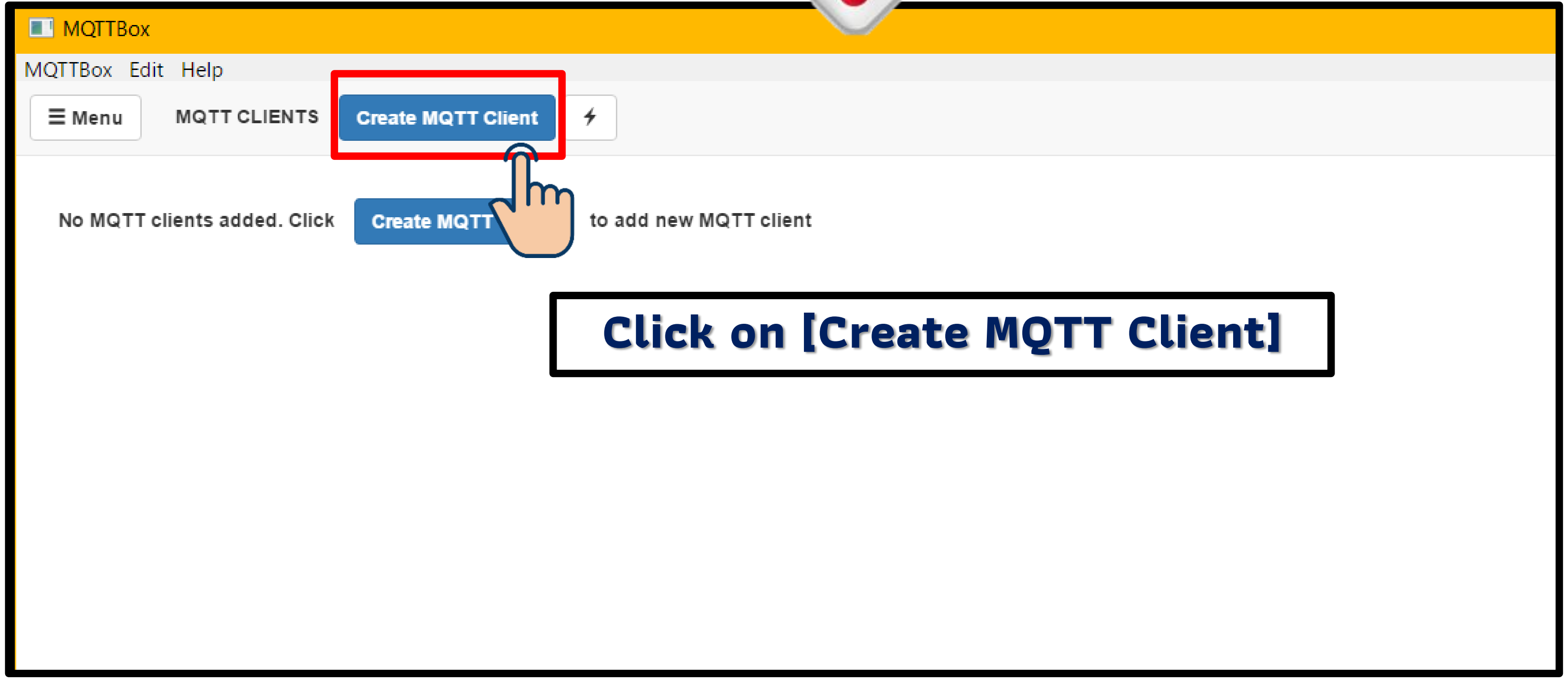
Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

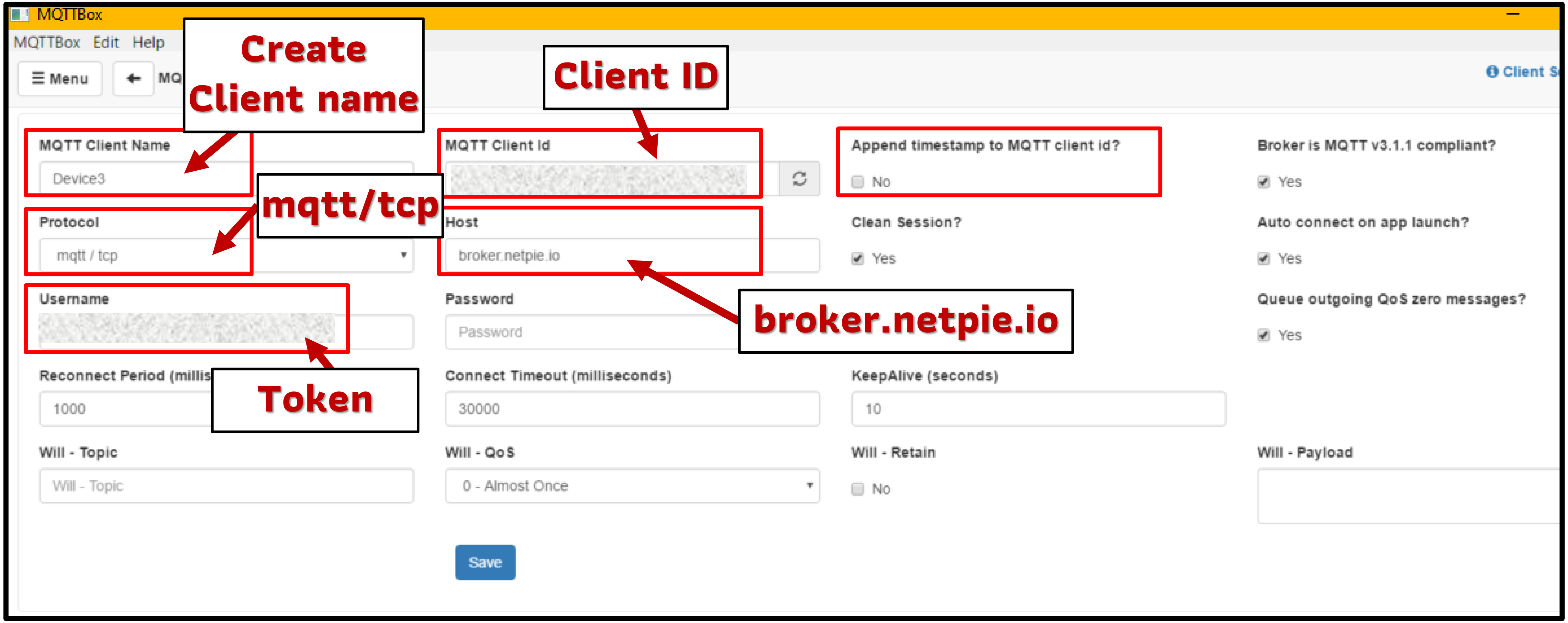
Exercise 3 : Communication on NETPIE2020 with Device Group

Open another MQTTBox window



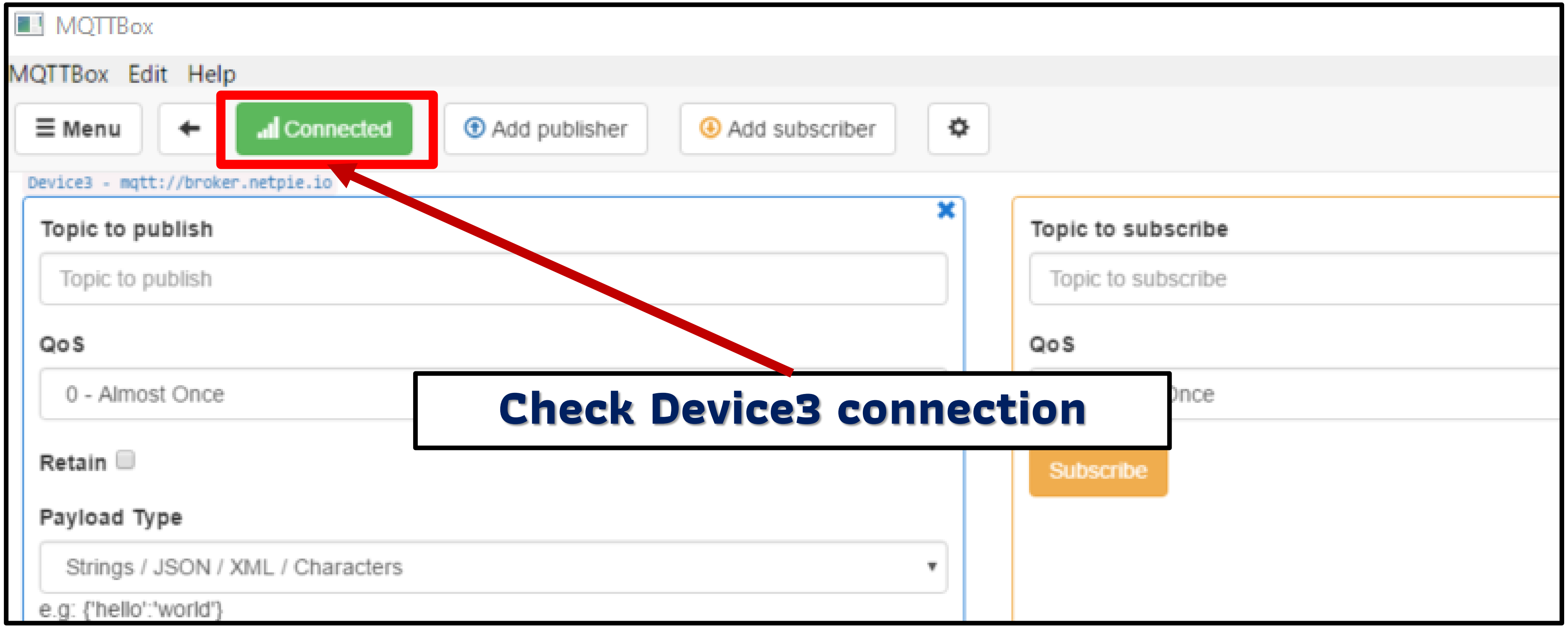
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



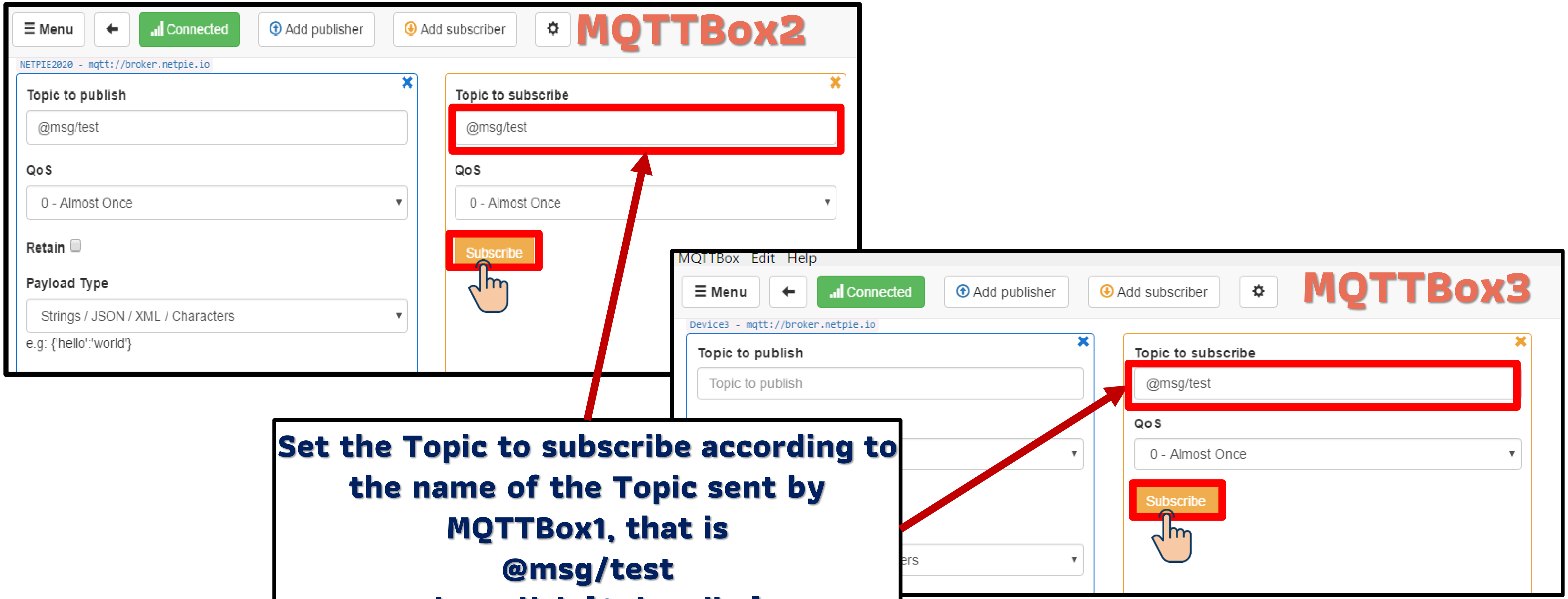
3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group



Set the Topic to subscribe according to the name of the Topic sent by MQTTBox1, that is @msg/test
Then click [Subscribe]

3 - Type of Communication in NETPIE2020

Exercise 3 : Communication on NETPIE2020 with Device Group

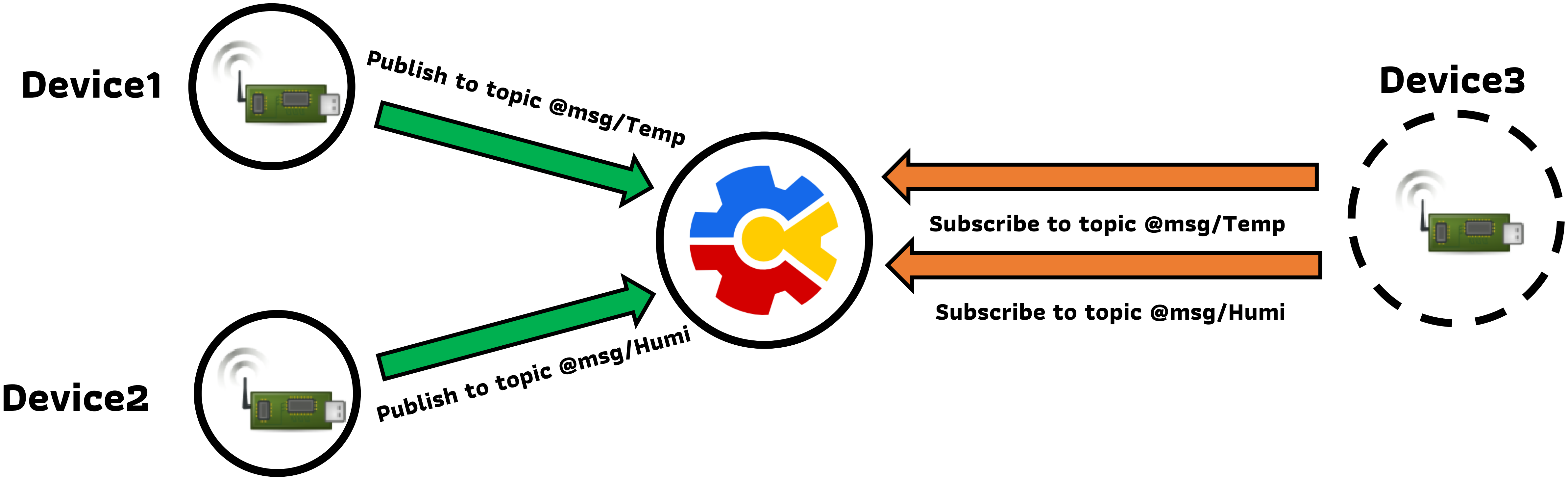
MQTTBox2 received the message sent by MQTTBox1 “Hello NETPIE2020”

MQTTBox3 did not receive message because it does not belong to the group

3 - Type of Communication in NETPIE2020

Wildcard Topic

If you want Device 3 to receive messages from Device1 and Device2, you need to subscribe to 2 topics

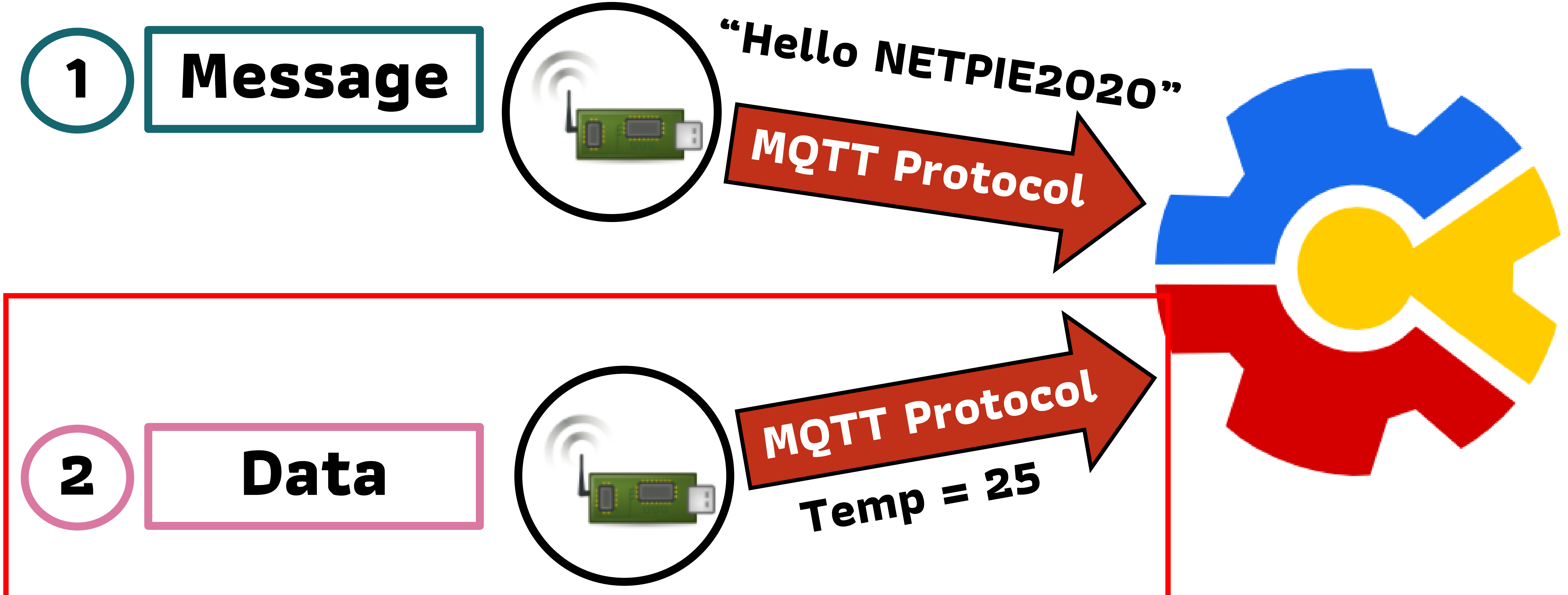


4

Data Management in NETPIE2020

4 - Data Management in NETPIE2020

There are two types of messages sent to NETPIE2020 via MQTT.



4 - Data Management in NETPIE2020

NETPIE2020 manages device information in 5 main sections.

1. Device Shadow : Latest device database
2. Device Schema : Device data structure
3. Device Feed : Device Data Storage
4. Device Trigger : Conditional device data
5. Event Hooks : Device transmission formats

We will talk about this on 11 Sep'21

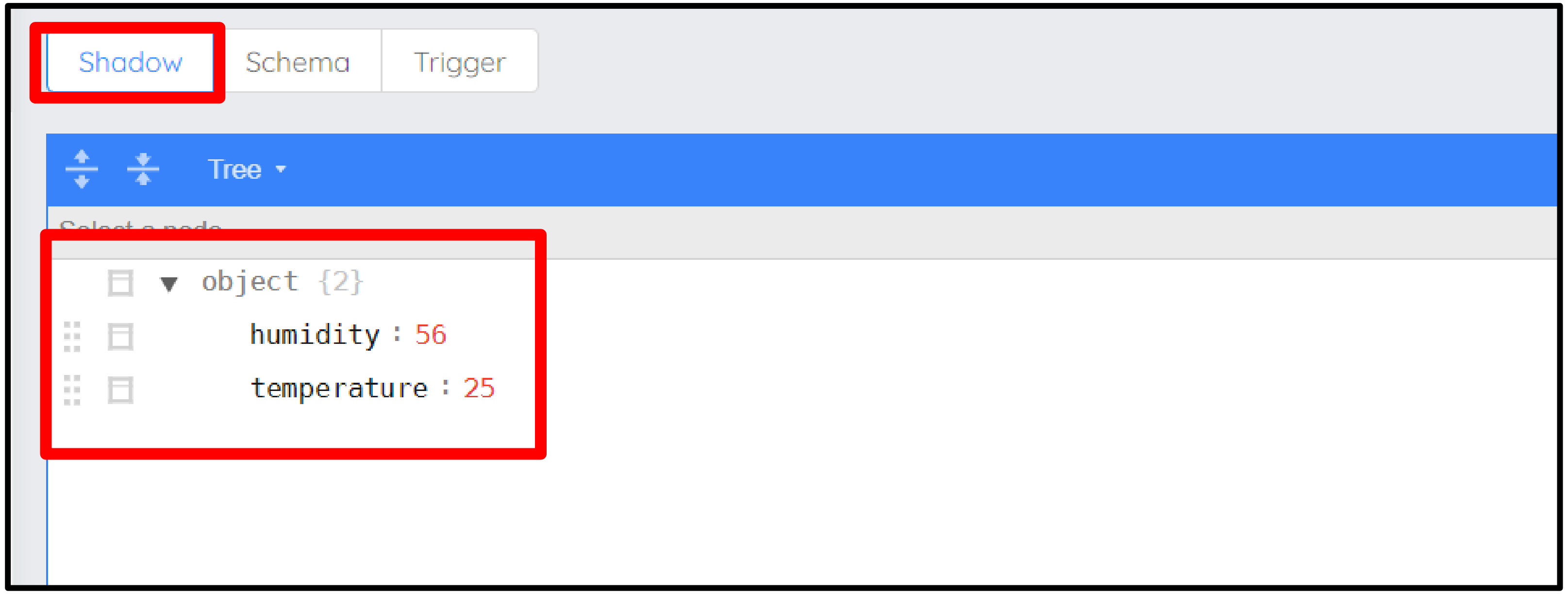
NETPIE2020

4 - Data Management in NETPIE2020

Device Shadow

Device Shadow is the virtual database of the device that is allocated for every device for two types of data storage

1. **Device Shadow Data : sensor data, for example**
2. **Device Shadow State : device online/offline status, for example**



4 - Data Management in NETPIE2020

Device Shadow

The MQTT topic that is involved with managing Device Shadow

Shadow Topic

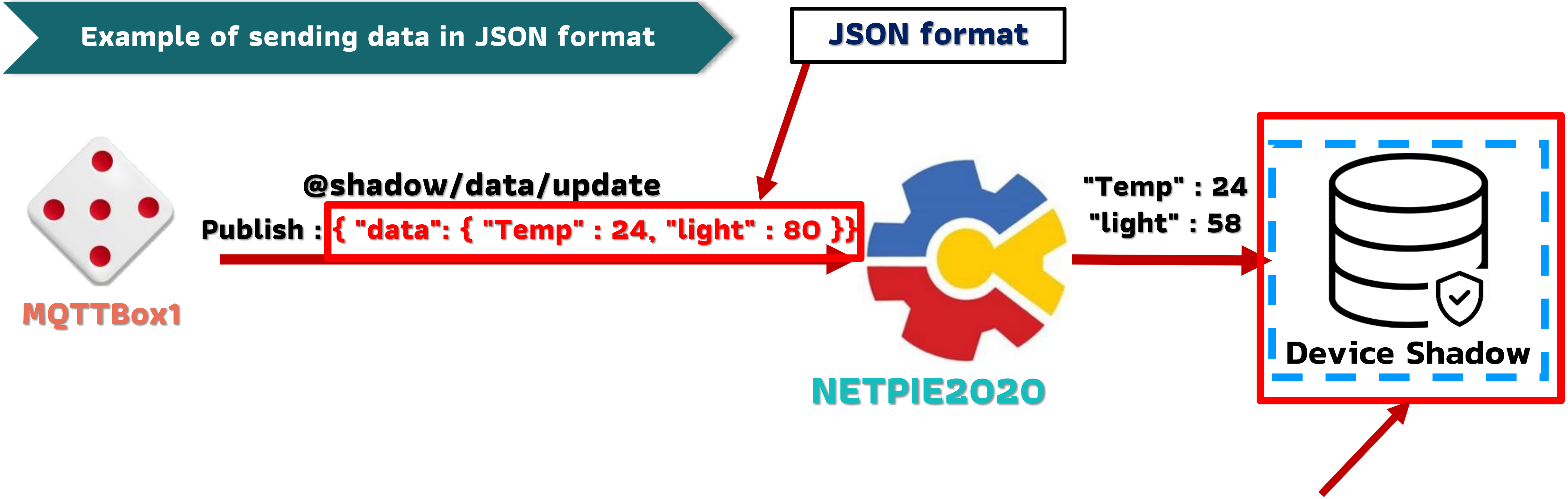
Used to manage own device shadow for publishing to edit Shadow information, and subscribing to receive Shadow information.

Publish Topic	Description	Subscribe Topic
@shadow/data/update	To update Shadow Data value by sending a payload in JSON format	**@shadow/data/updated**

4 - Data Management in NETPIE2020

Device Shadow

Example of sending data in JSON format



But in order to store data in Device Shadow, you need to create a Device Schema first.

4 - Data Management in NETPIE2020

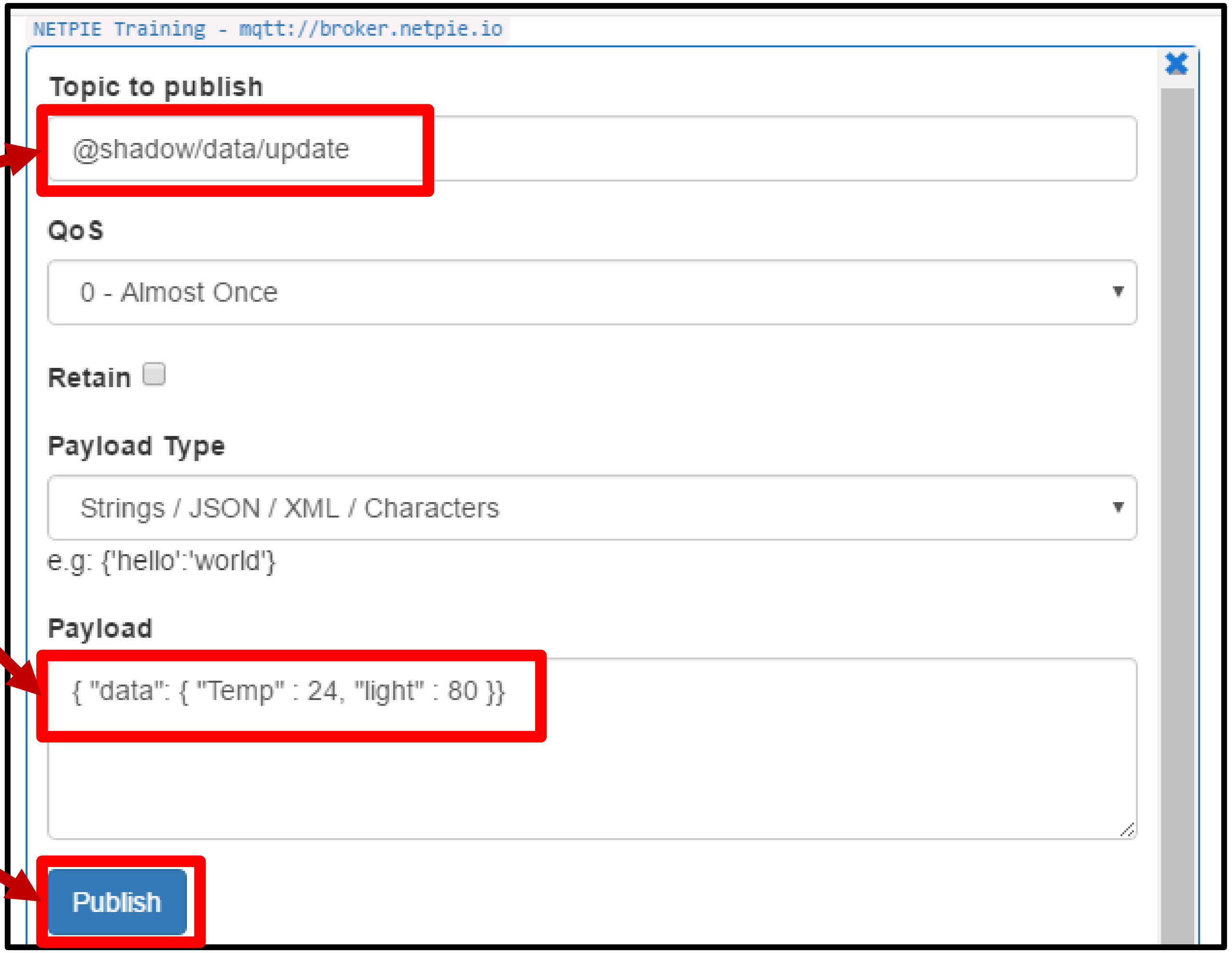
Exercise 4 : Sent data to NETPIE

Open MQTTBox1 window

Topic to send data

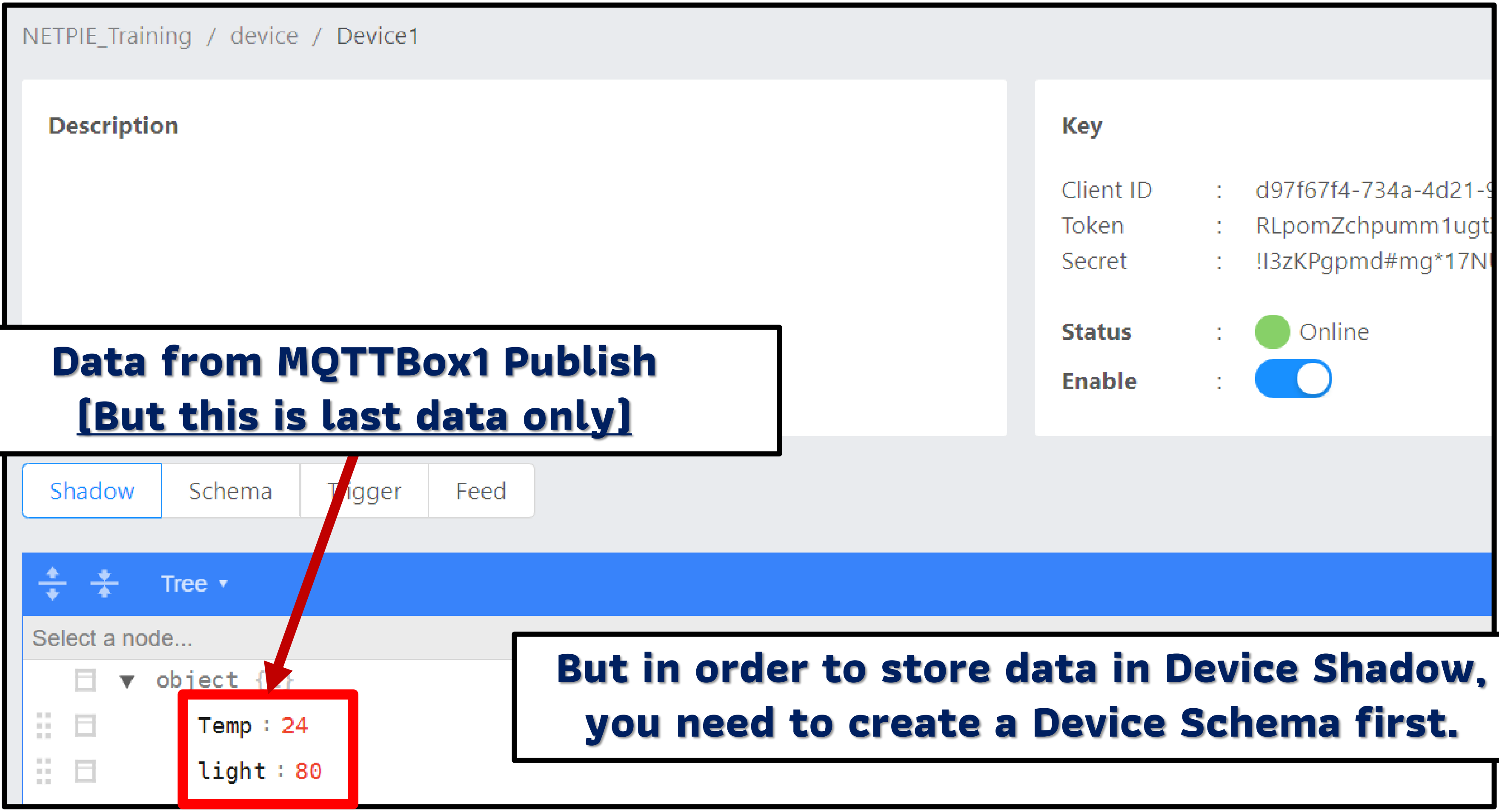
Set payload data such as
{ "data": { "Temp" : 24, "light" : 80 }}

Click Publish for sent data



4 - Data Management in NETPIE2020

Exercise 4 : Sent data to NETPIE



Data from MQTTBox1 Publish
[But this is last data only]

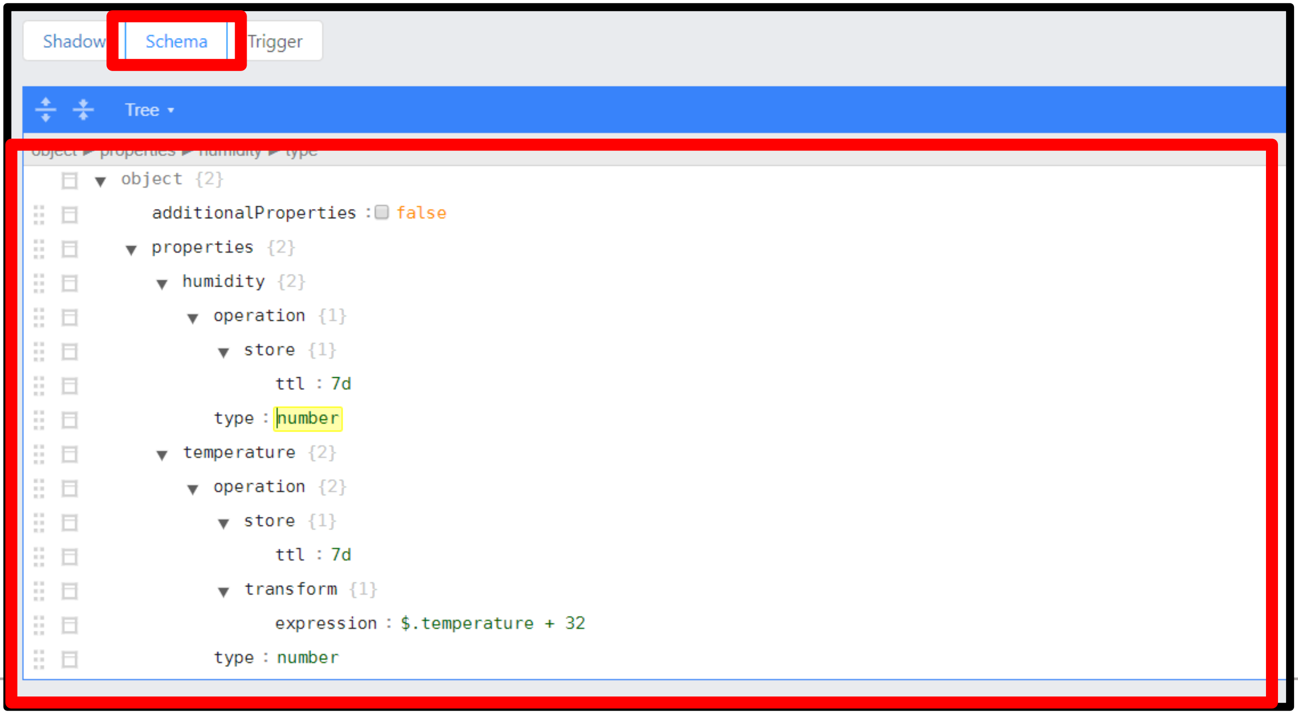
But in order to store data in Device Shadow, you need to create a Device Schema first.

4 - Data Management in NETPIE2020

Device Schema

Device Schema is a data structure defined to manage Device Shadow. For devices that need data management, a Device Schema should be created. This Device Schema allows the server to

- **Check data types before storing**
- **Converting data before storage, such as changing data units.**
- **Data collection in Timeseries Database (Feed)**



4 - Data Management in NETPIE2020

Device Schema

Declaring the Device Schema in JSON format.

```
{
 "additionalProperties": false,
 "properties": {
  "light": {
   "operation": {
    "store": {
     "ttl": "7d"
    }
   },
   "type": "number"
  },
  "temperature": {
   "operation": {
    "store": {
     "ttl": "7d"
    },
    "transform": {
     "expression": [[$.temperature]*1.8] + 32"
    }
   },
   "type": "number"
  }
 }
}
```

additionalProperties

This is the permission to save data to Shadow or Timeseries Database in case the data is not declared in the properties section. additionalProperties has 2 status values

true : authorize data write to Shadow or Timeseries Database

false : prohibit data write for data not defined in Properties

In the example, two items in properties are humidity and temperature

If the data received are temp, humid, light

addtionalProperties = true : will record temperature, humidity and light

addtionalProperties = false : will record only temperature, light

4 - Data Management in NETPIE2020

Device Schema

Declaring the Device Schema in JSON format

```
{
 "additionalProperties": false,
 "properties": {
   "light": {
    "operation": {
      "store": {
        "ttl": "7d"
      }
    },
    "type": "number"
   },
   "temperature": {
    "operation": {
      "store": {
        "ttl": "7d"
      },
      "transform": {
        "expression": [($.temperature)*1.8) + 32"
      }
    },
    "type": "number"
   }
  }
}
```

Properties

First, define a field name (for example, 'light' and 'Temperature') and define properties for each field, which are divided into two parts:

Operation For setting up data handling in that field, including

store for keeping data inTimeseries Database

ttl duration of data inTimeseries Database. Data that exceeds expiration date will be automatically deleted. To store system data, this value needs to be set in units of ms (milliseconds), s (seconds), m (minutes), h (hours), d (days), y (years)

Transform data transformation before keeping

expression formula for transformation

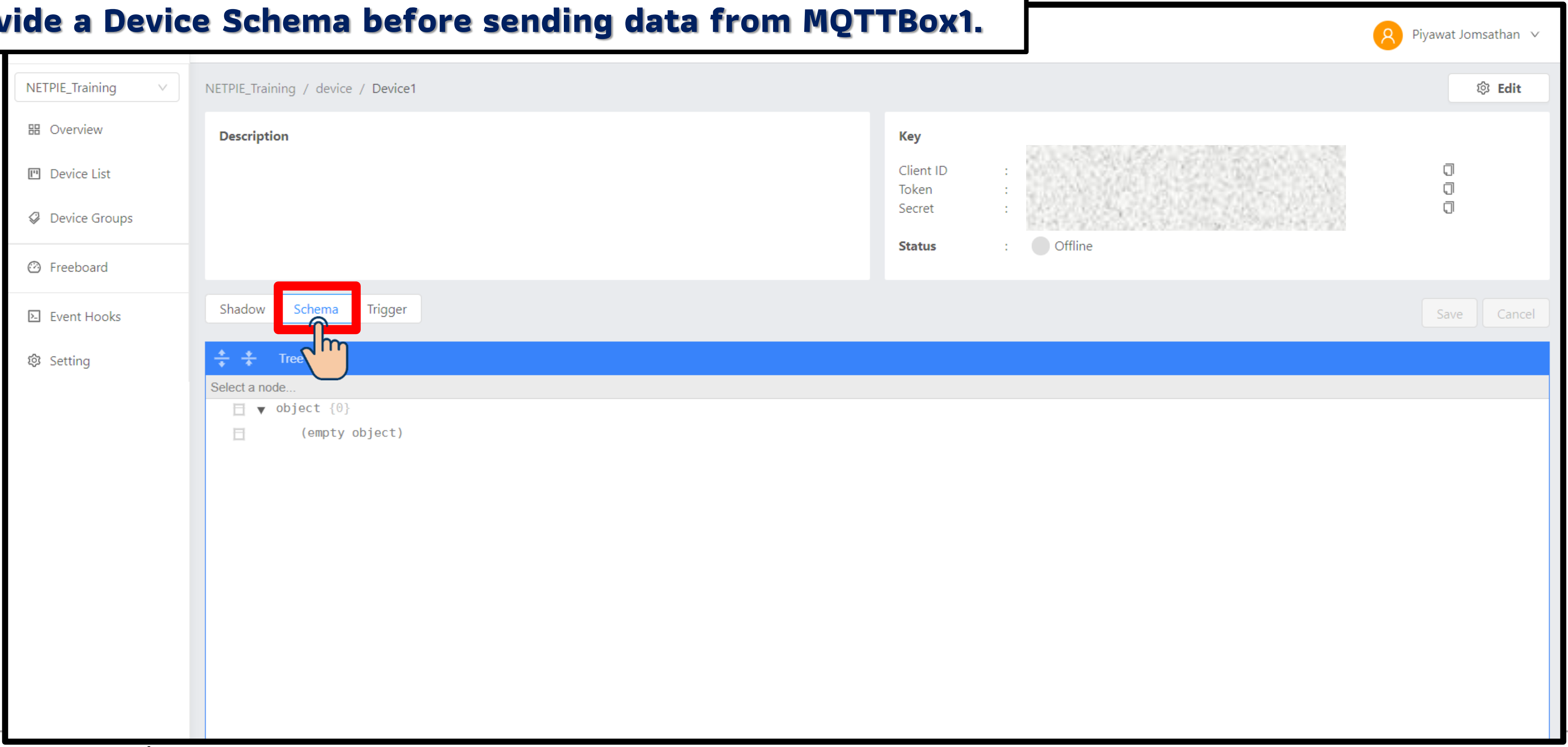
For example, a formula to convert Celsius to Fahrenheit= ($.temperature*1.8) + 32

Type data type in a field, such as number, string, array, object

4 - Data Management in NETPIE2020

Exercise 5 : Set Schema and Send Data to Device Feed

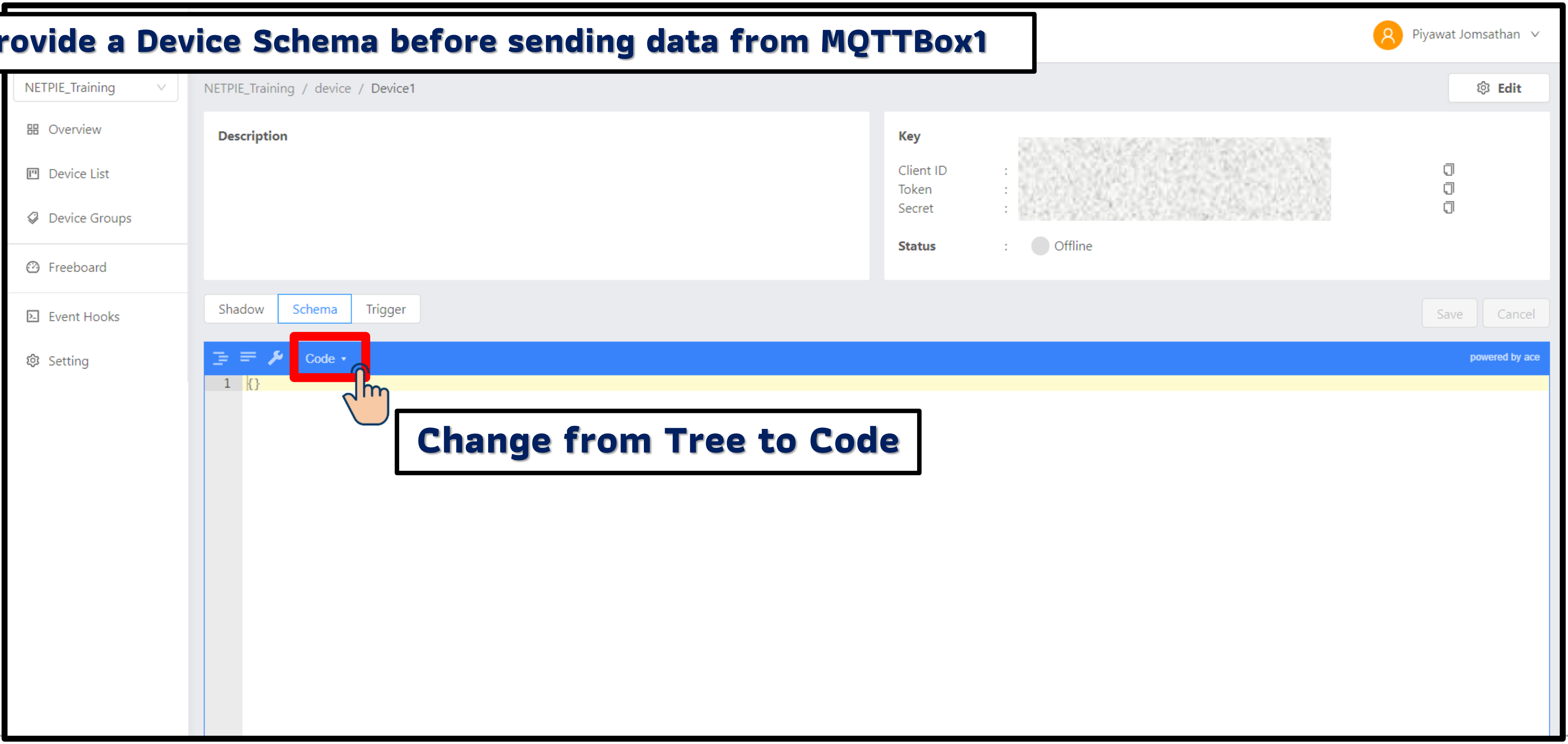
Provide a Device Schema before sending data from MQTTBox1.



4 - Data Management in NETPIE2020

Exercise 5 : Set Schema and Send Data to Device Feed

Provide a Device Schema before sending data from MQTTBox1



Change from Tree to Code

4 - Data Management in NETPIE2020

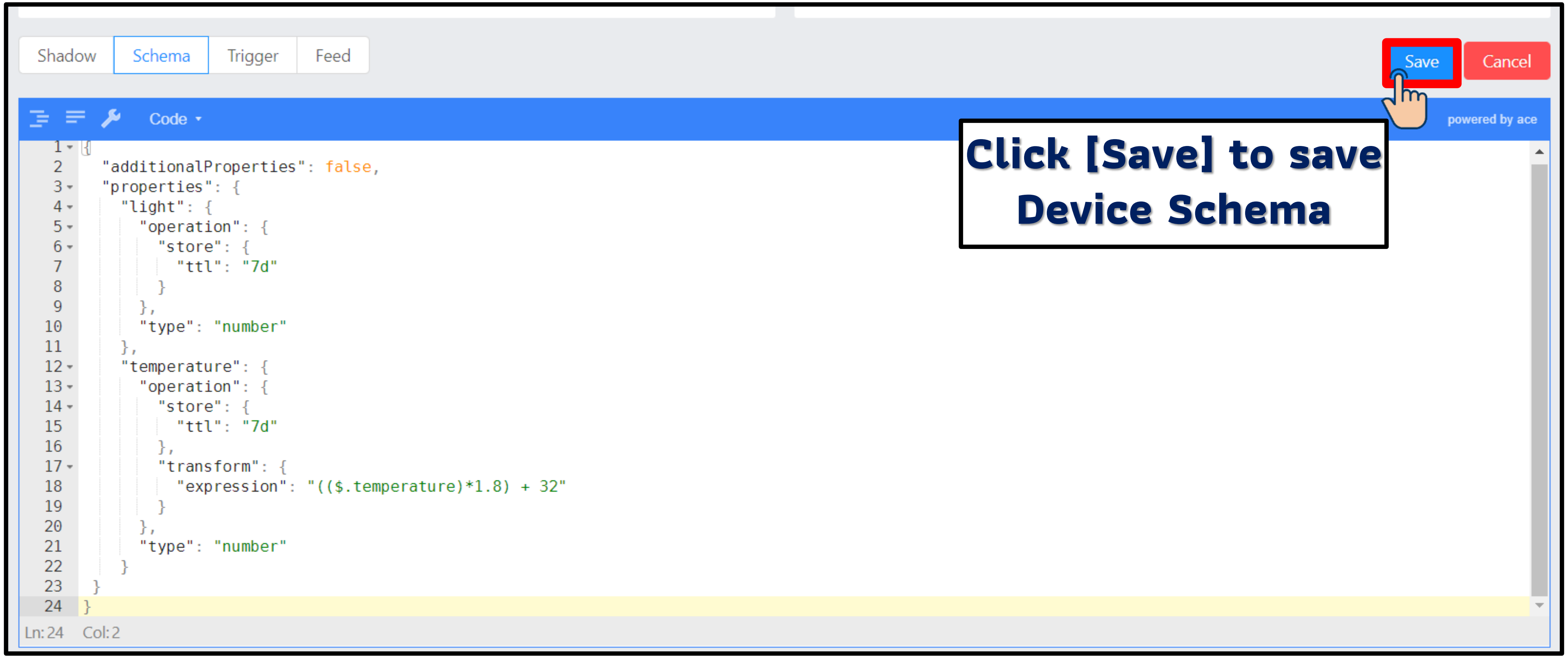
Exercise 5 : Set Schema and Send Data to Device Feed

Device Schema in JSON format

```
{
  "additionalProperties": false,
  "properties": {
    "light": {
      "operation": {
        "store": {
          "ttl": "7d"
        }
      },
      "type": "number"
    },
    "temperature": {
      "operation": {
        "store": {
          "ttl": "7d"
        },
        "transform": {
          "expression": "[($.temperature)*1.8] + 32"
        }
      },
      "type": "number"
    }
  }
}
```

Copy to Device Schema

Exercise 5 : Set Schema and Send Data to Device Feed



4 - Data Management in NETPIE2020

Exercise 5 : Set Schema and Send Data to Device Feed

Device Schema in JSON format

Set **additionalProperties = false**
In order not to save values other than listed in properties to Shadow.

The first variable is **light**
with properties
- Store for 7 days
- Data type is number

```
{
 "additionalProperties": false,
 "properties": {
  "light": {
   "operation": {
    "store": {
     "ttl": "7d"
    }
   },
   "type": "number"
  },
  "temperature": {
   "operation": {
    "store": {
     "ttl": "7d"
    },
    "transform": {
     "expression": "[($.temperature)*1.8] + 32"
    }
   },
   "type": "number"
  }
 }
}
```

4 - Data Management in NETPIE2020

Exercise 5 : Set Schema and Send Data to Device Feed

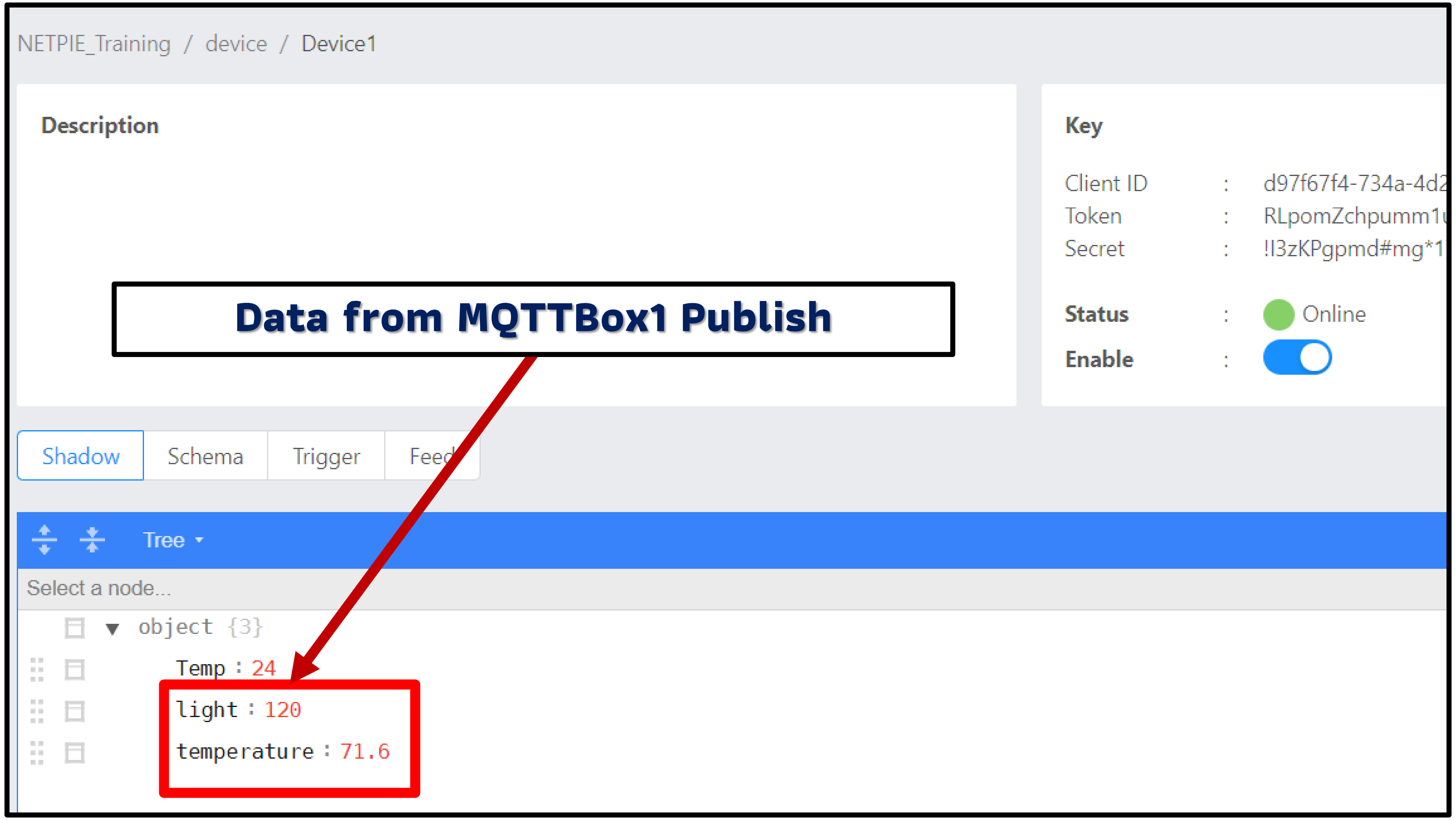
Device Schema in JSON format

```
{
  "additionalProperties": false,
  "properties": {
    "light": {
      "operation": {
        "store": {
          "ttl": "7d"
        }
      },
      "type": "number"
    },
    "temperature": {
      "operation": {
        "store": {
          "ttl": "7d"
        },
        "transform": {
          "expression": "[($.temperature)*1.8] + 32"
        }
      },
      "type": "number"
    }
  }
}
```

Variable to store is **temperature** with properties

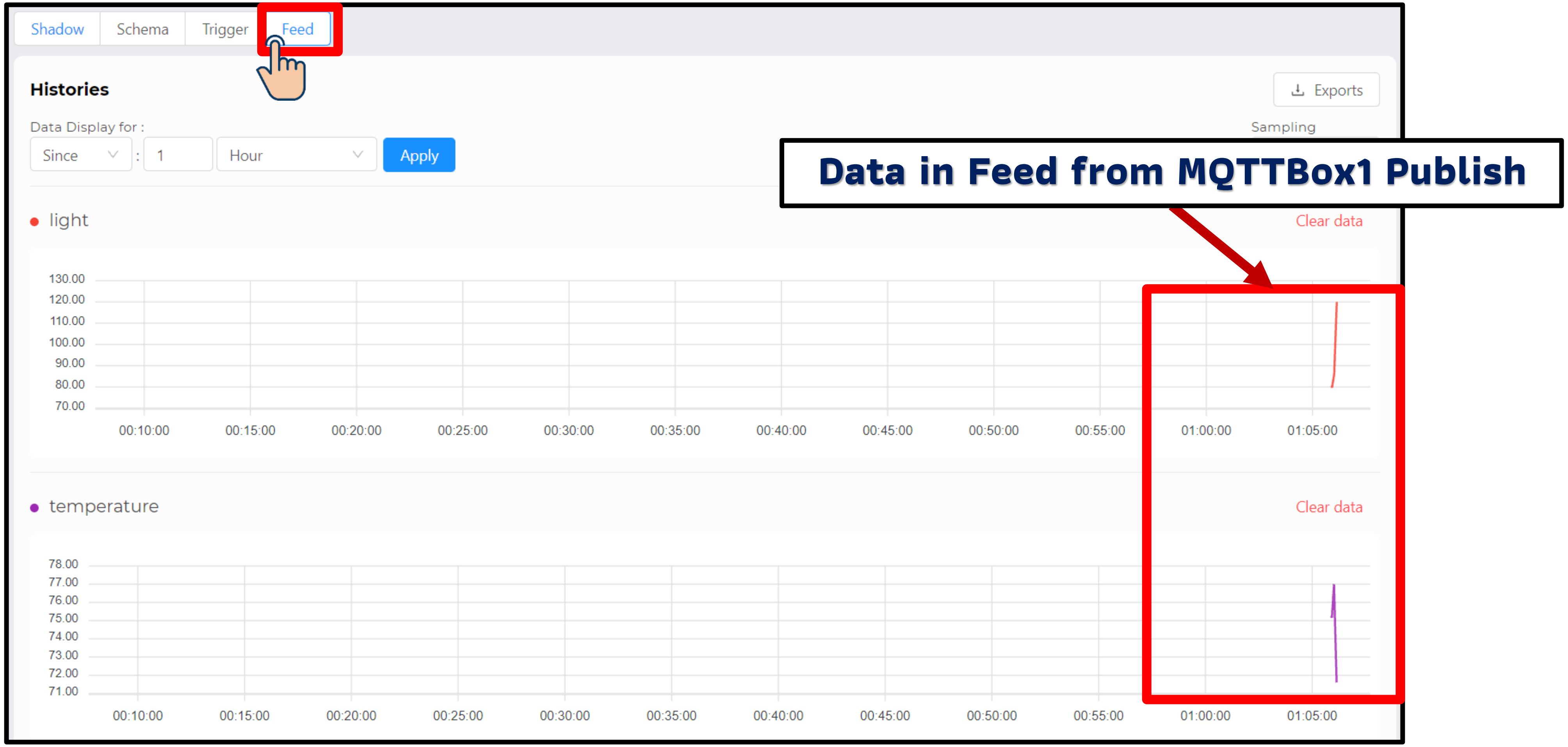
- Store for 7 days
- Convert from Celsius to Fahrenheit
- Variable type is number

Exercise 5 : Set Schema and Send Data to Device Feed



4 - Data Management in NETPIE2020

Exercise 5 : Set Schema and Send Data to Device Feed



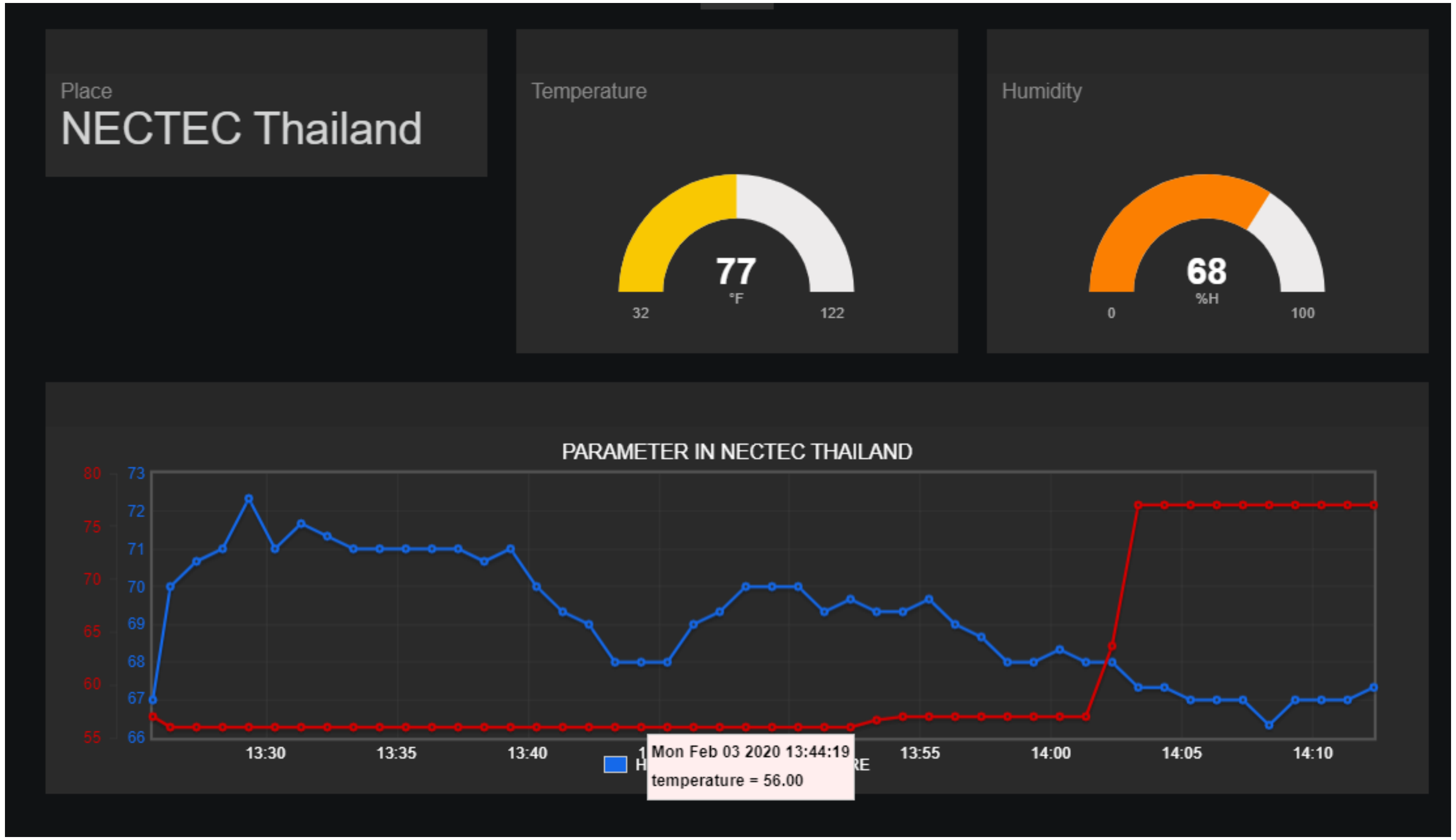
5 Freeboard in NETPIE2020

NETPIE
where things chat

5 - Freeboard in NETPIE2020

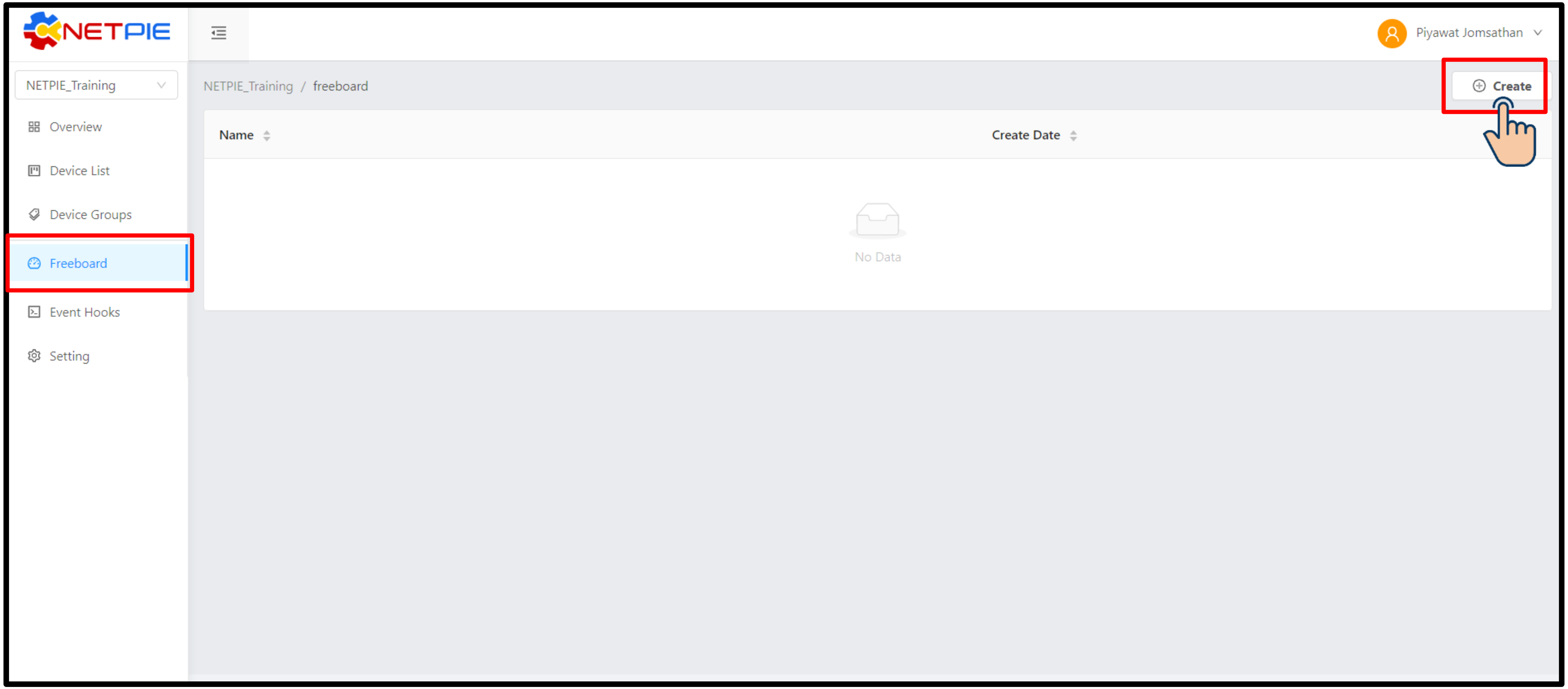
What is Freeboard?

NETPIE2020 Freeboard is a software panel for control and display data retrieved from Device Shadow. A developer creates and customizes Widget Plugins to suit user requirements, such as gauges, sliders, control buttons, and put Javascript commands for various actions.



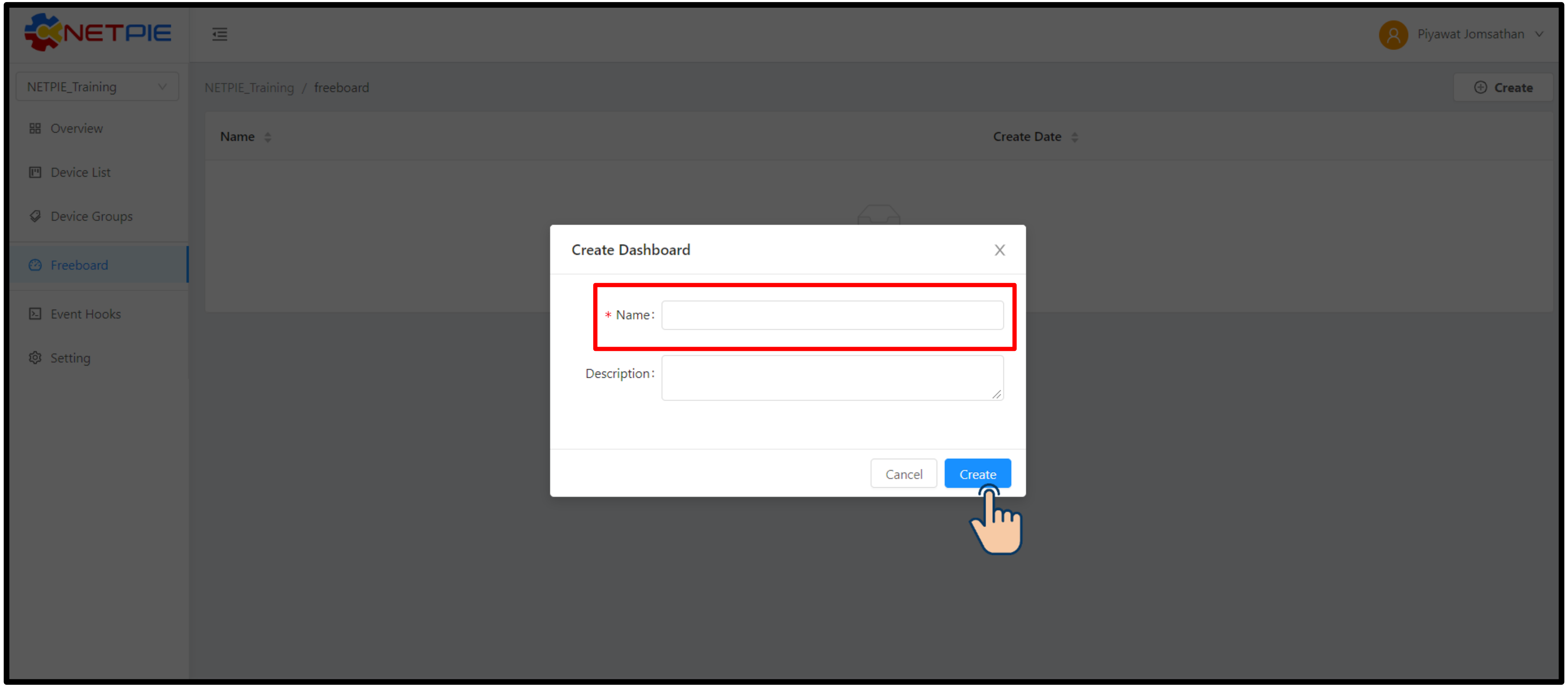
5 - Freeboard in NETPIE2020

Create a Freeboard



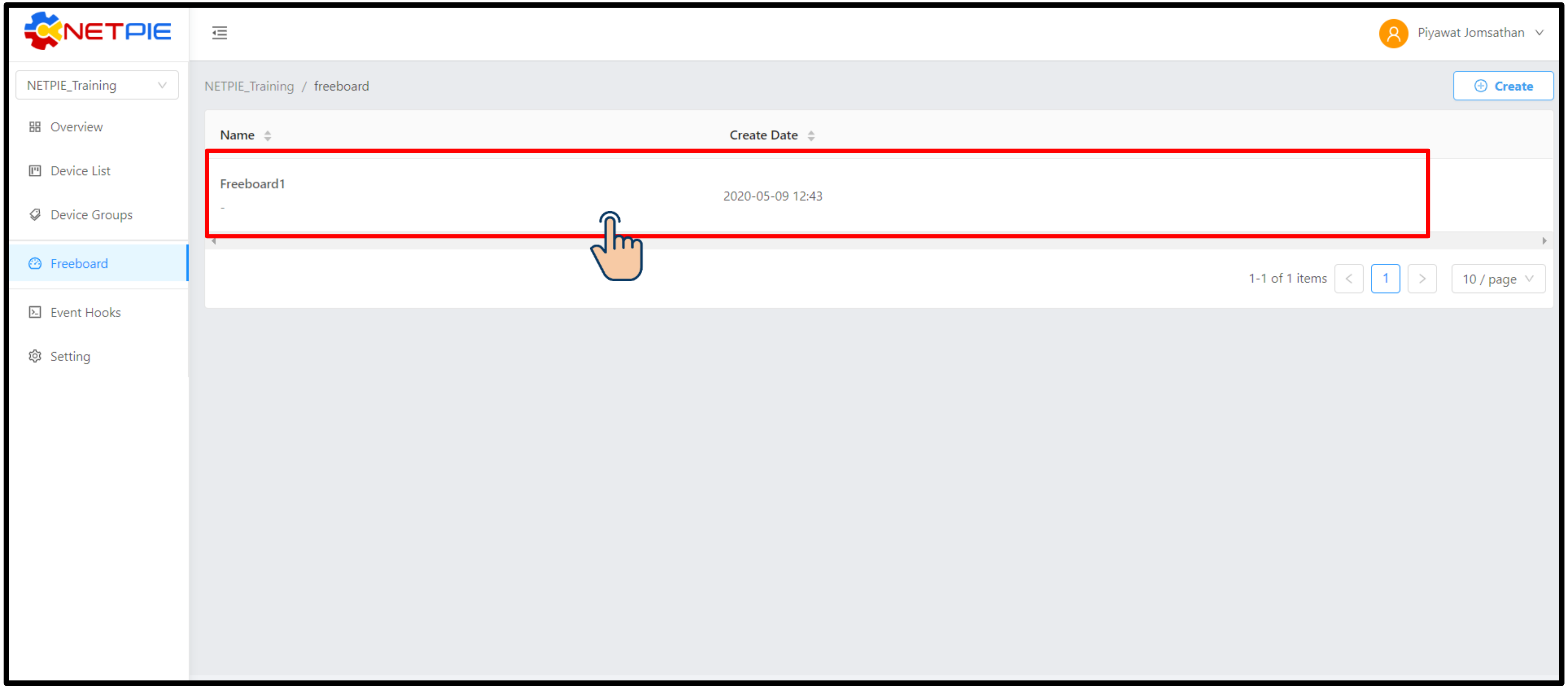
5 - Freeboard in NETPIE2020

Create a Freeboard



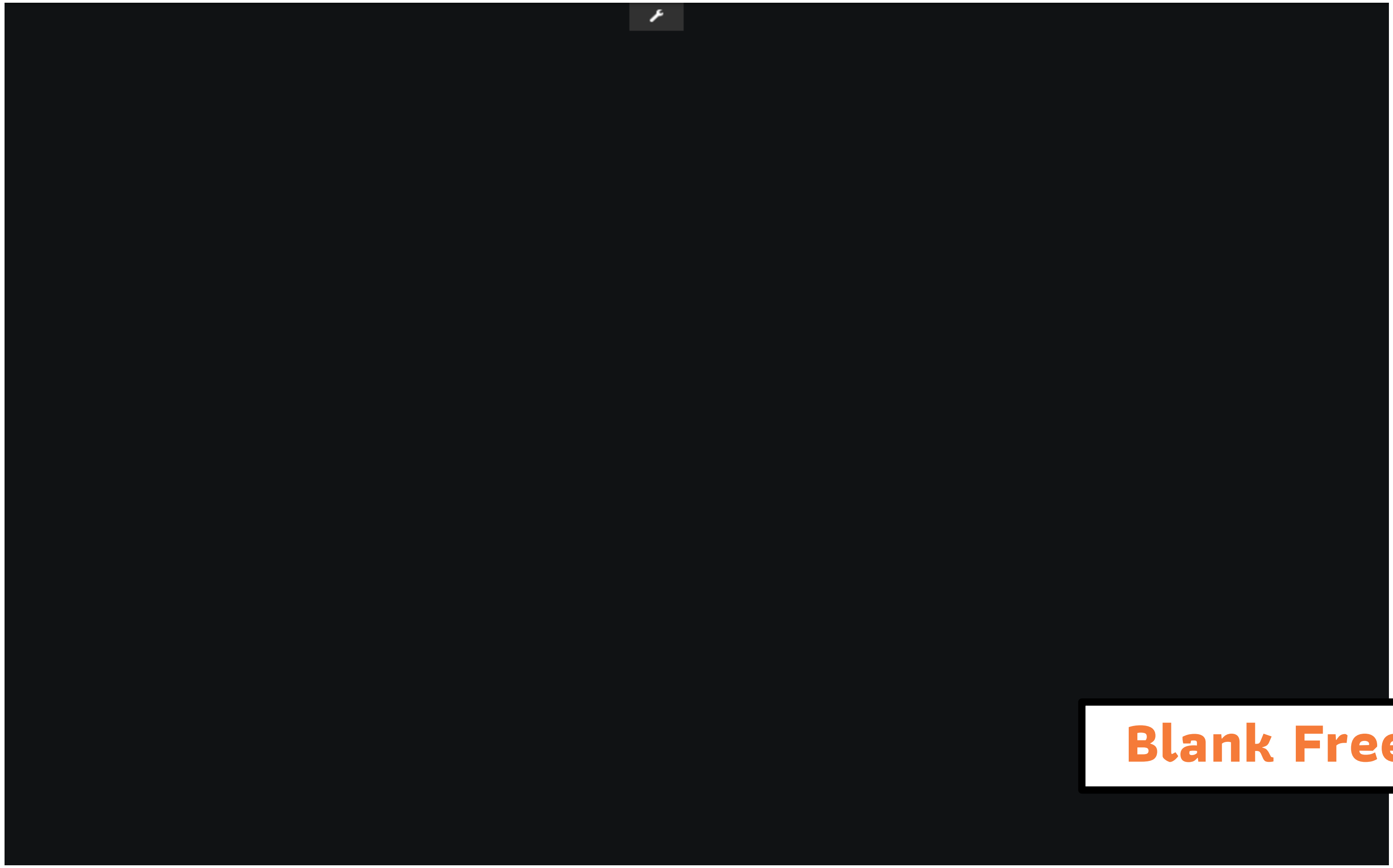
5 - Freeboard in NETPIE2020

Create a Freeboard



5 - Freeboard in NETPIE2020

Create a Freeboard



Blank Freeboard window

5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

@shadow/data/update

Publish : { "data": { "temperature" : 24, "light" : 80, "place" : "NECTEC" }}

MQTTBox1

NETPIE2020

"temperature" : 75.2
"light" : 80
"place: "NECTEC"

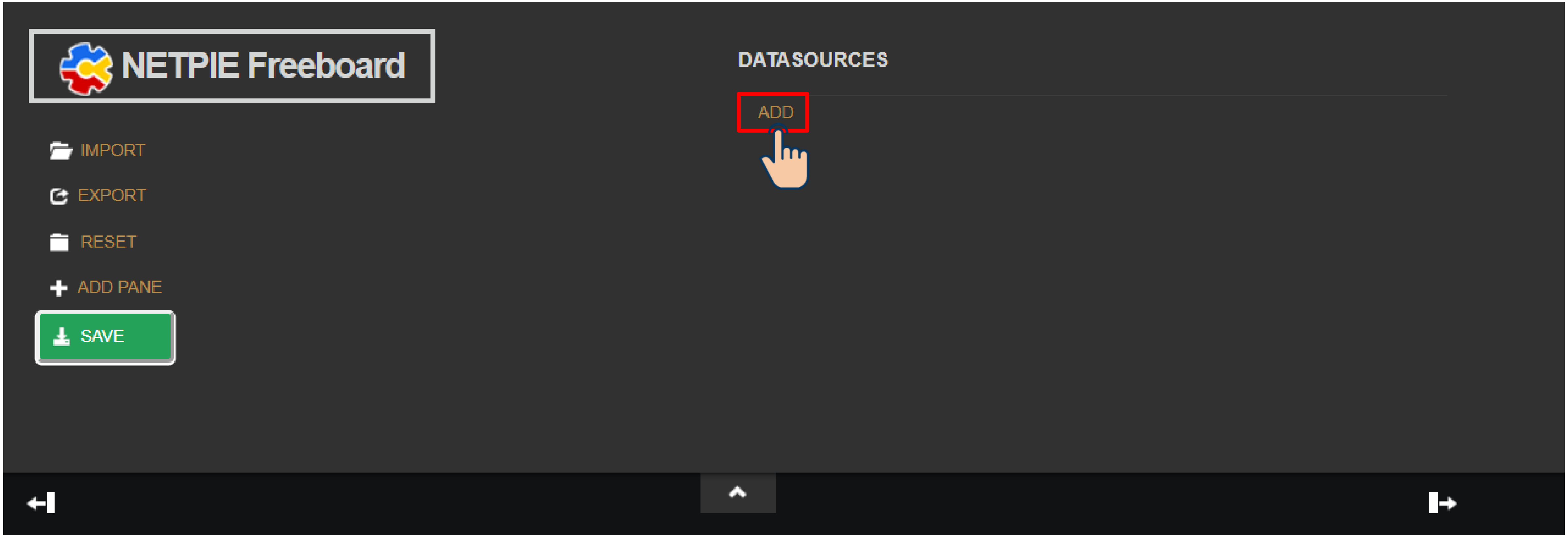
Device Shadow

5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

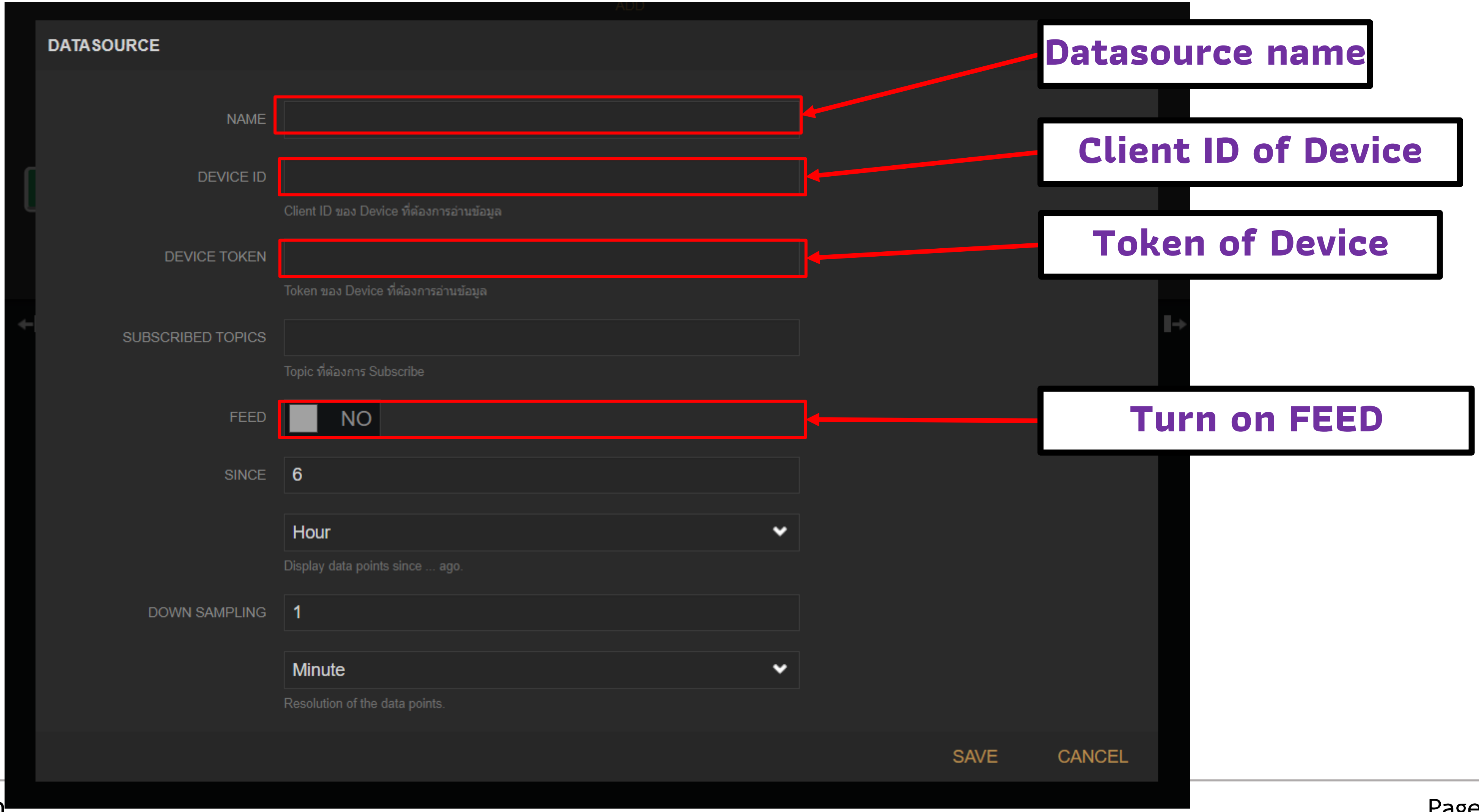
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



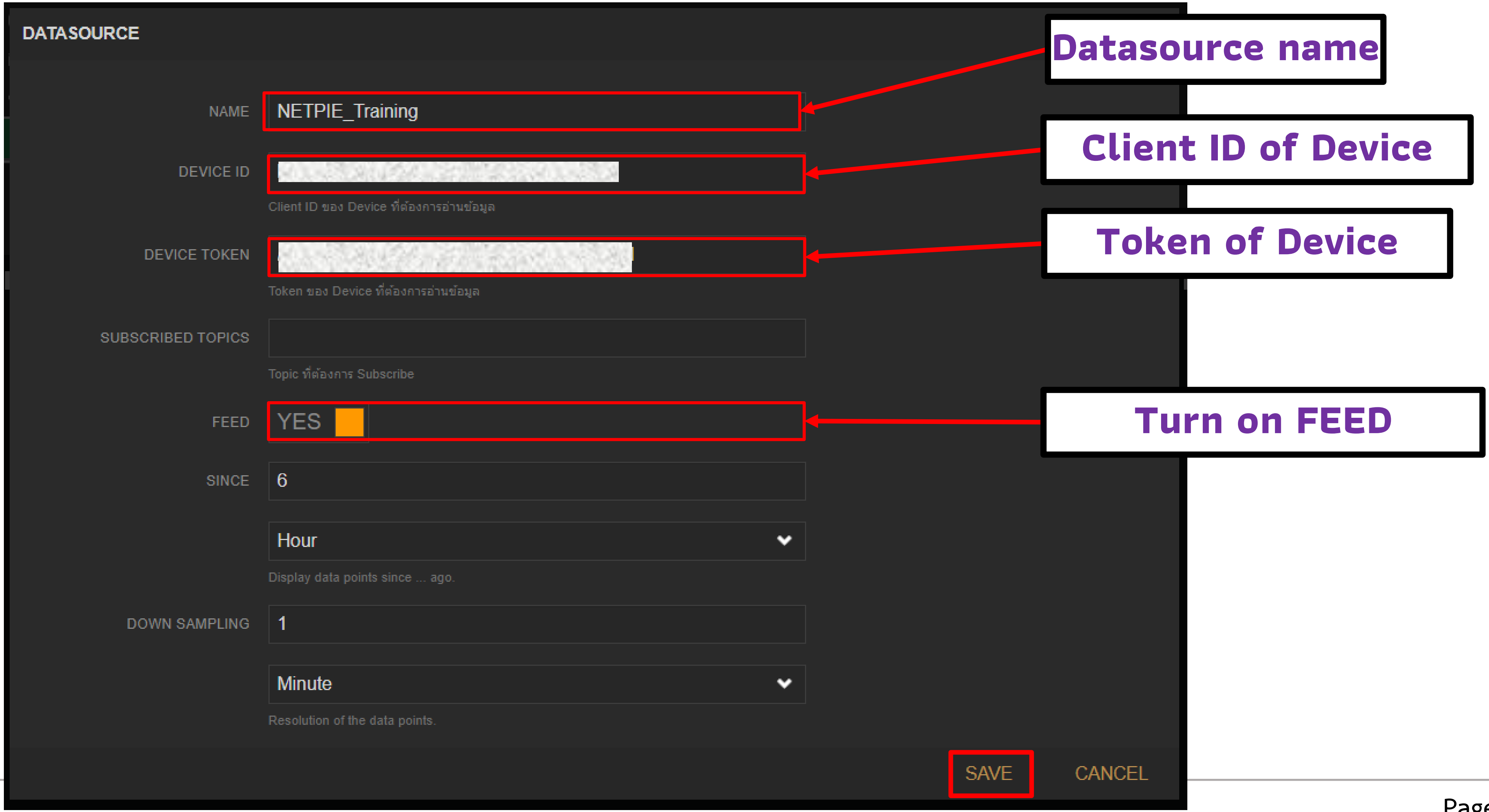
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



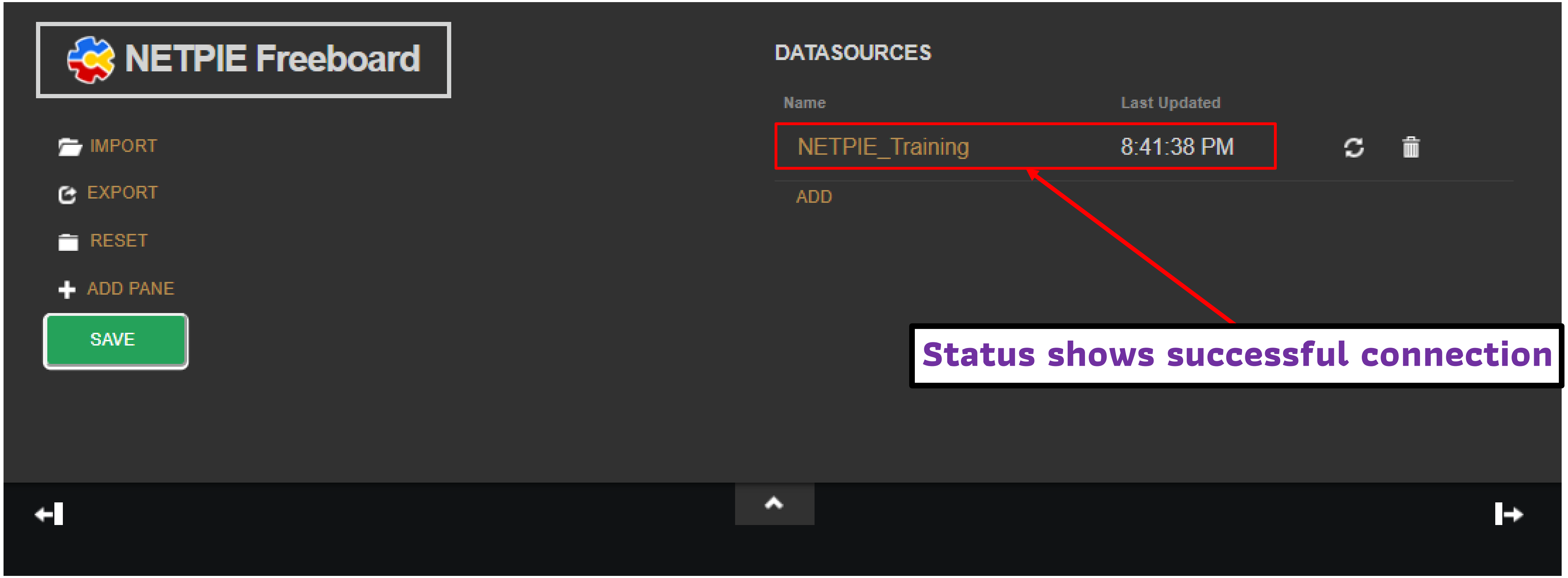
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



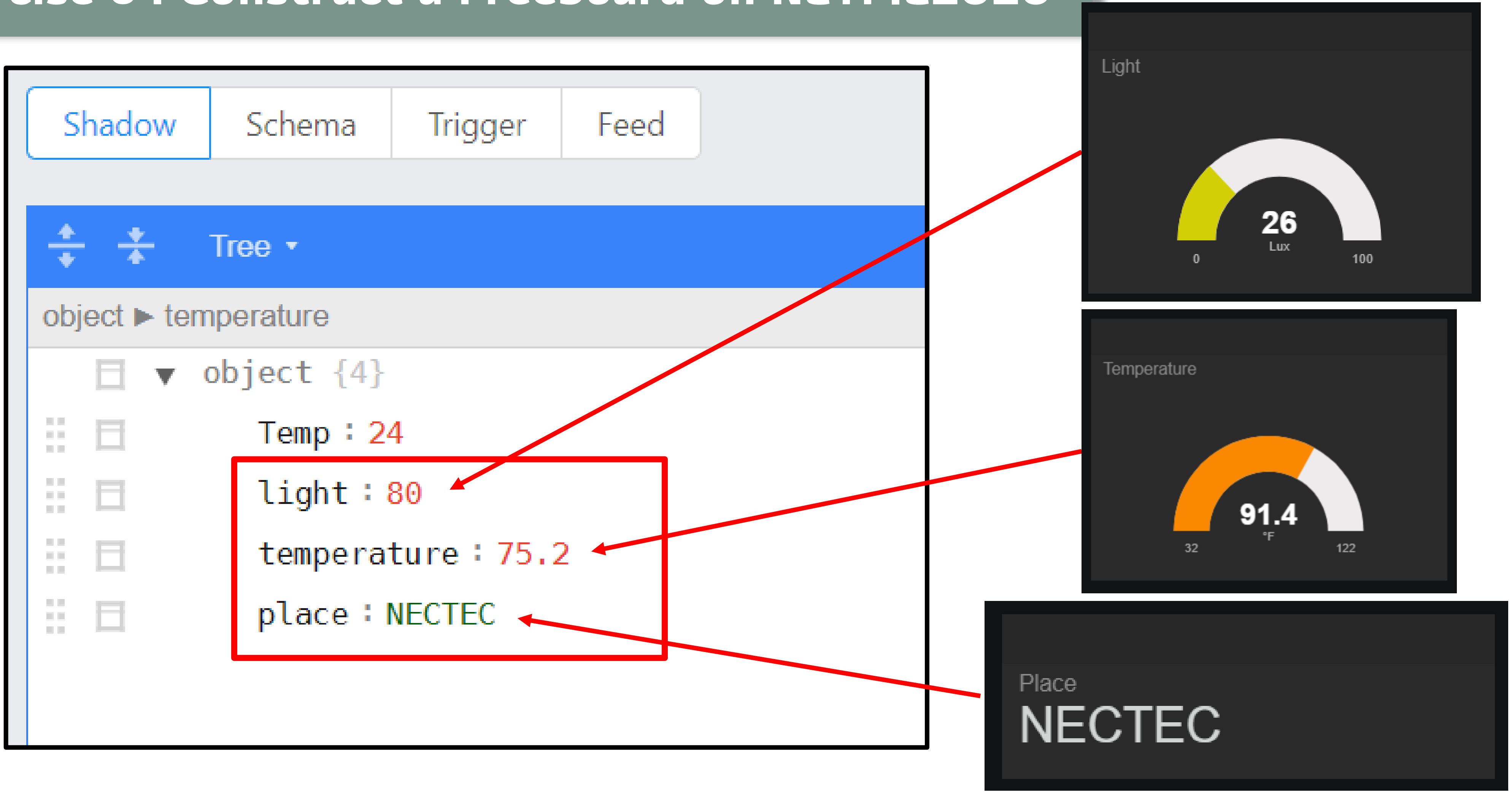
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



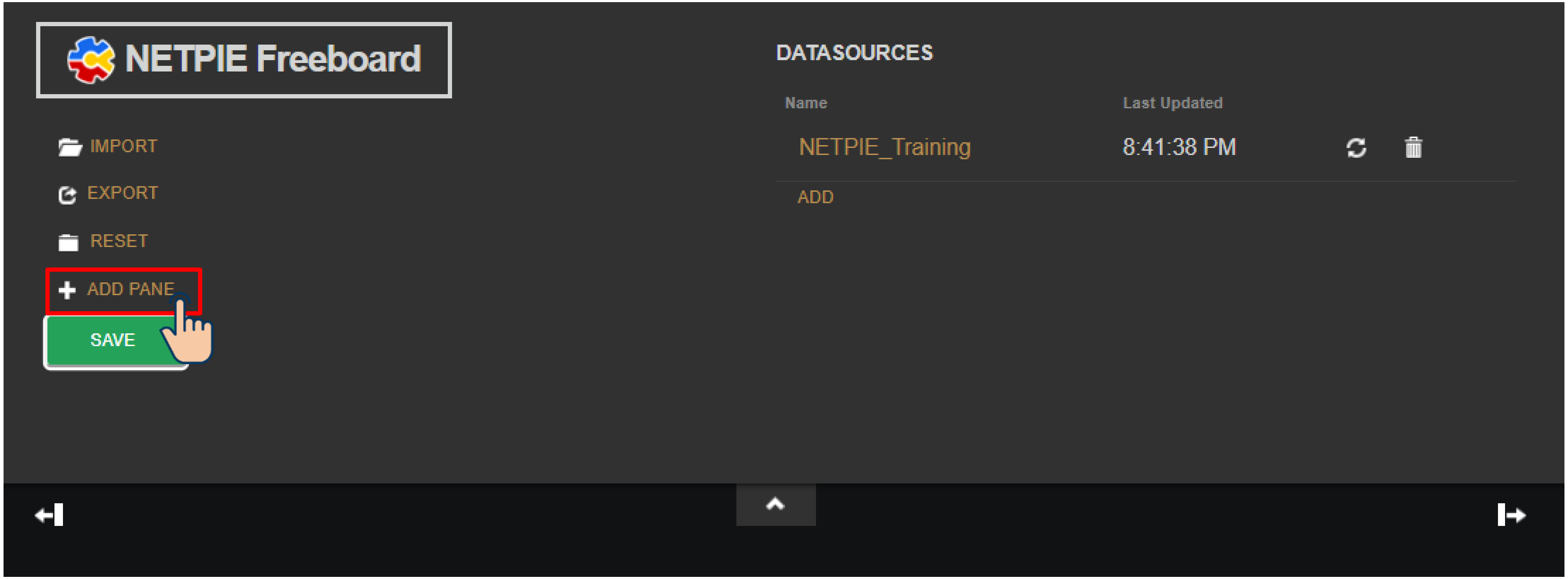
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



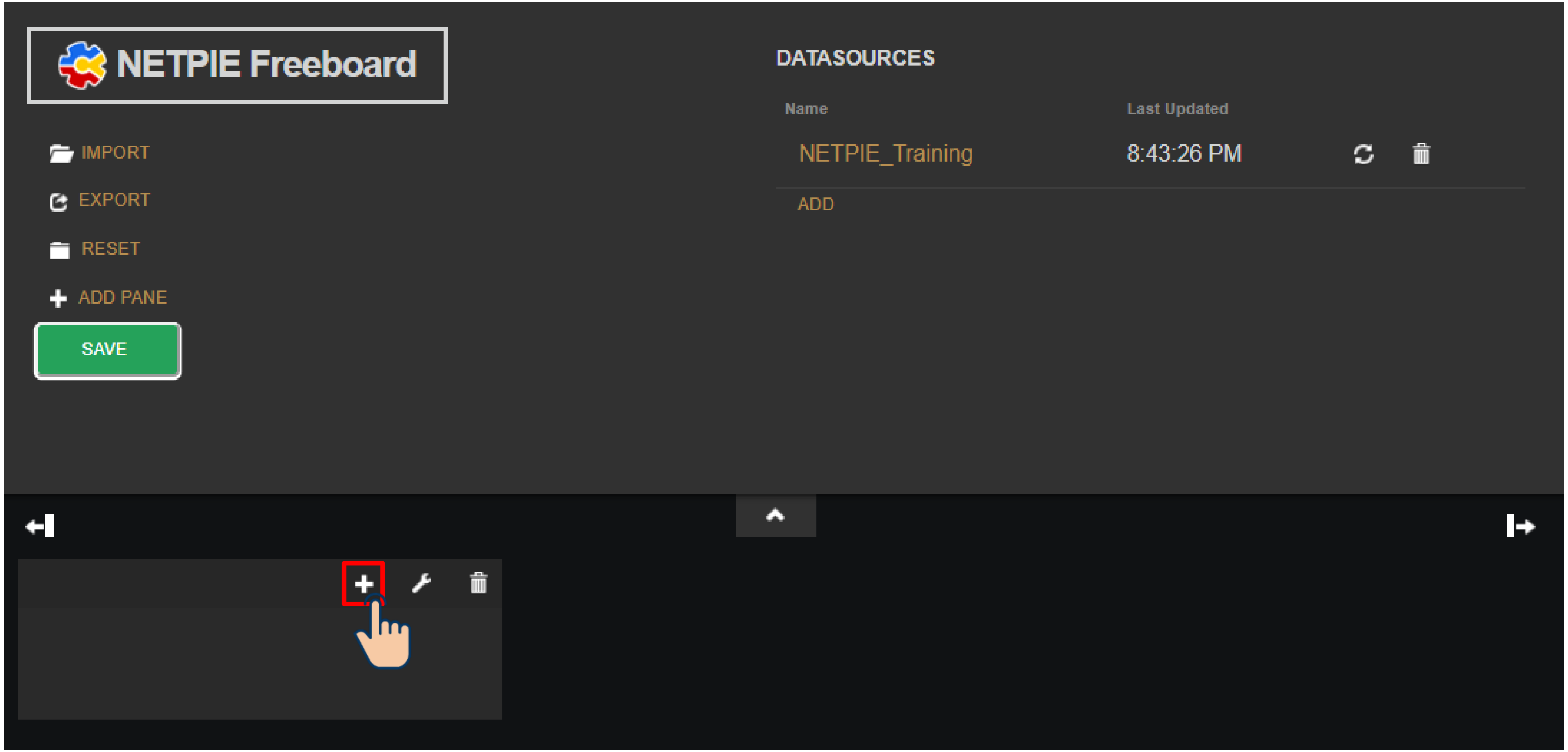
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

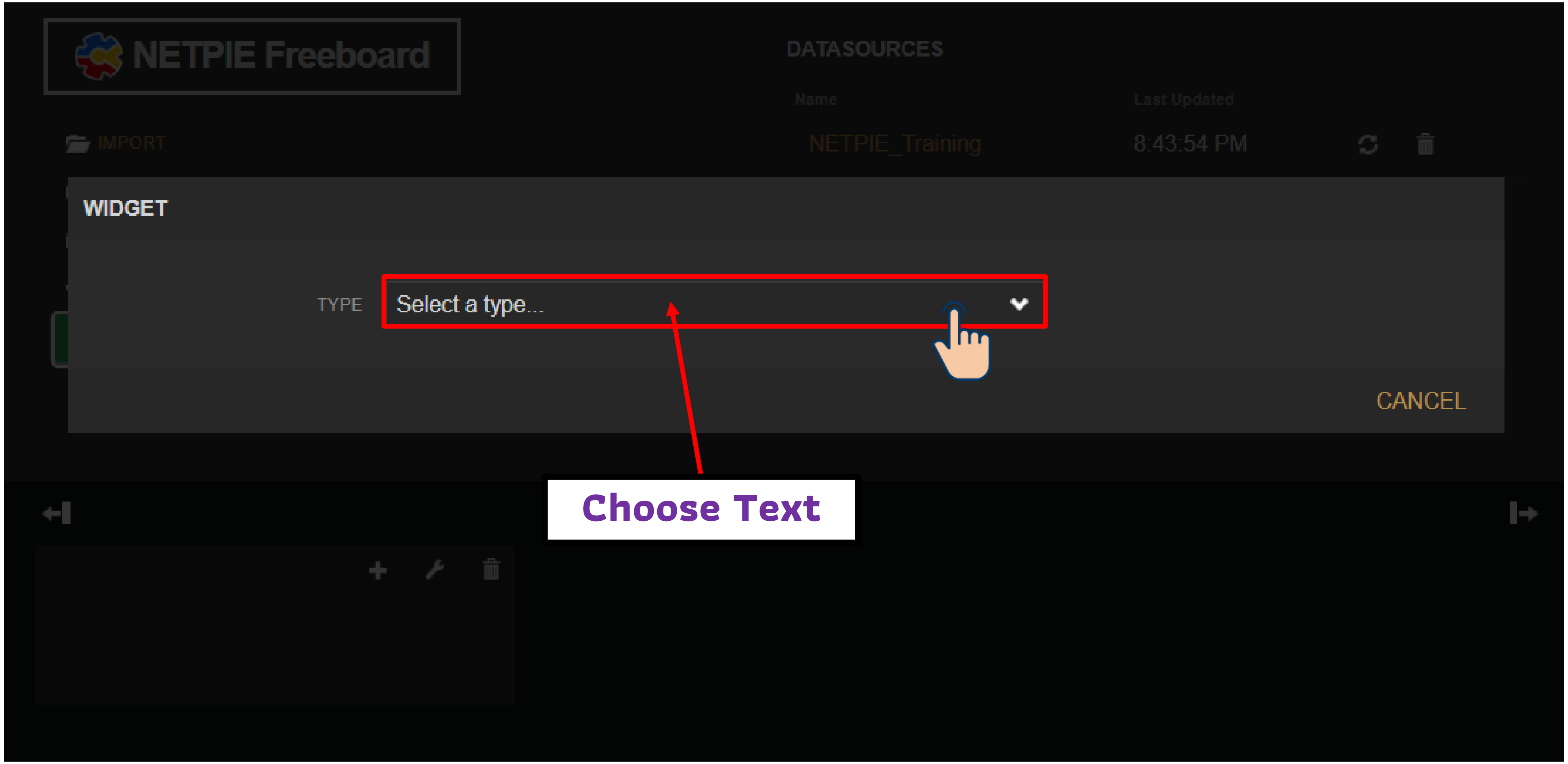


5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

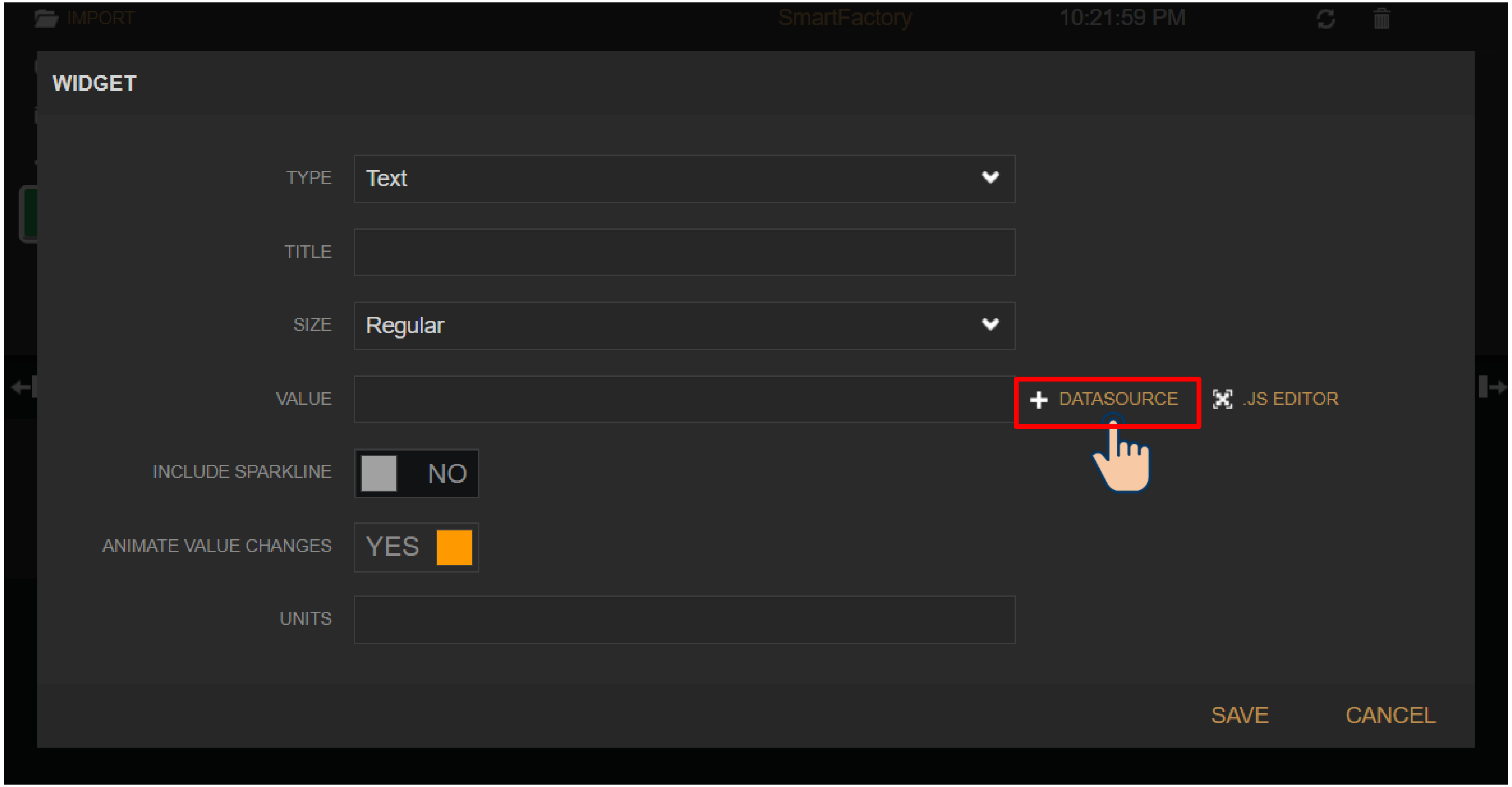


Exercise 6 : Construct a Freeboard on NETPIE2020



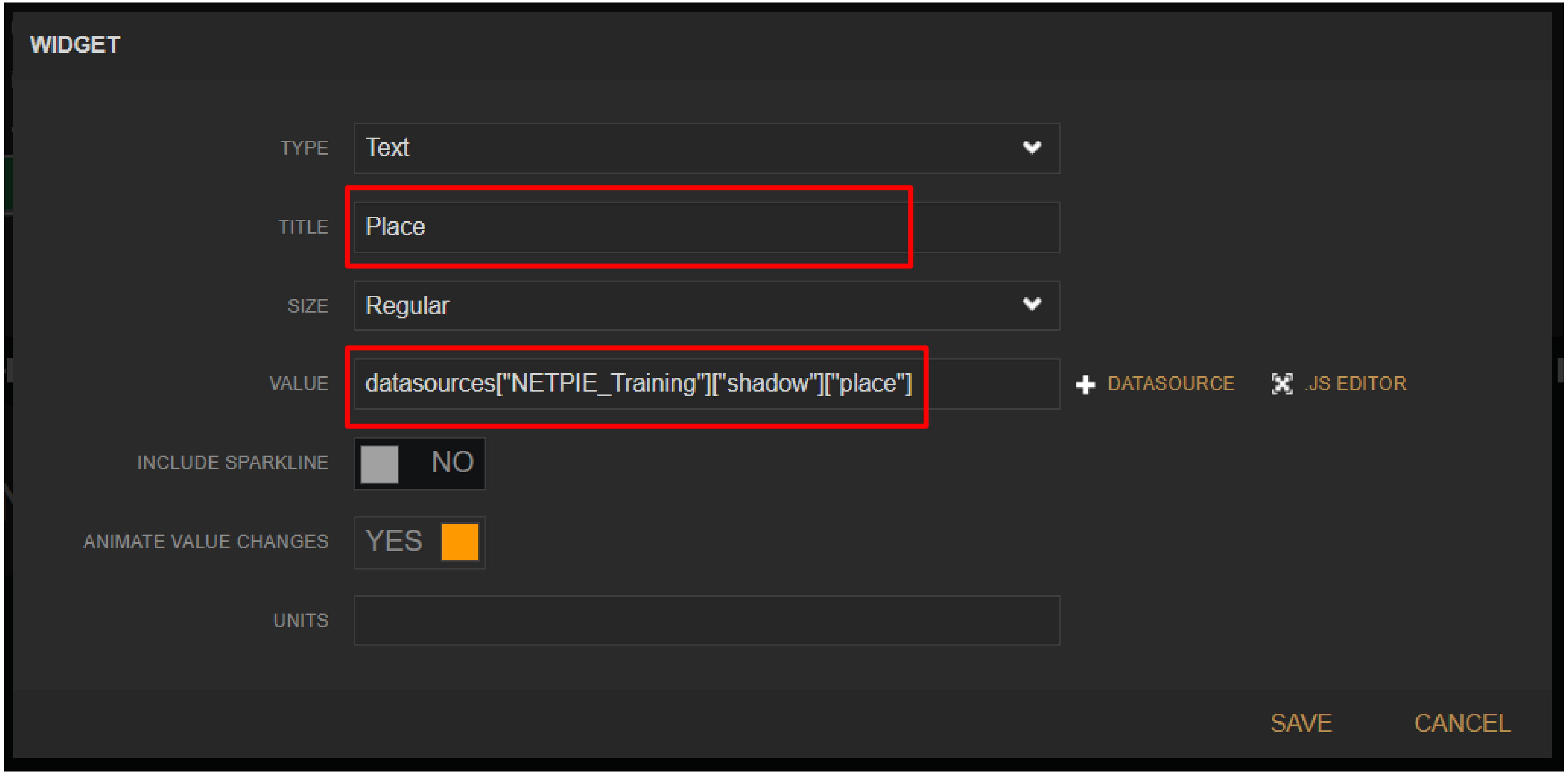
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



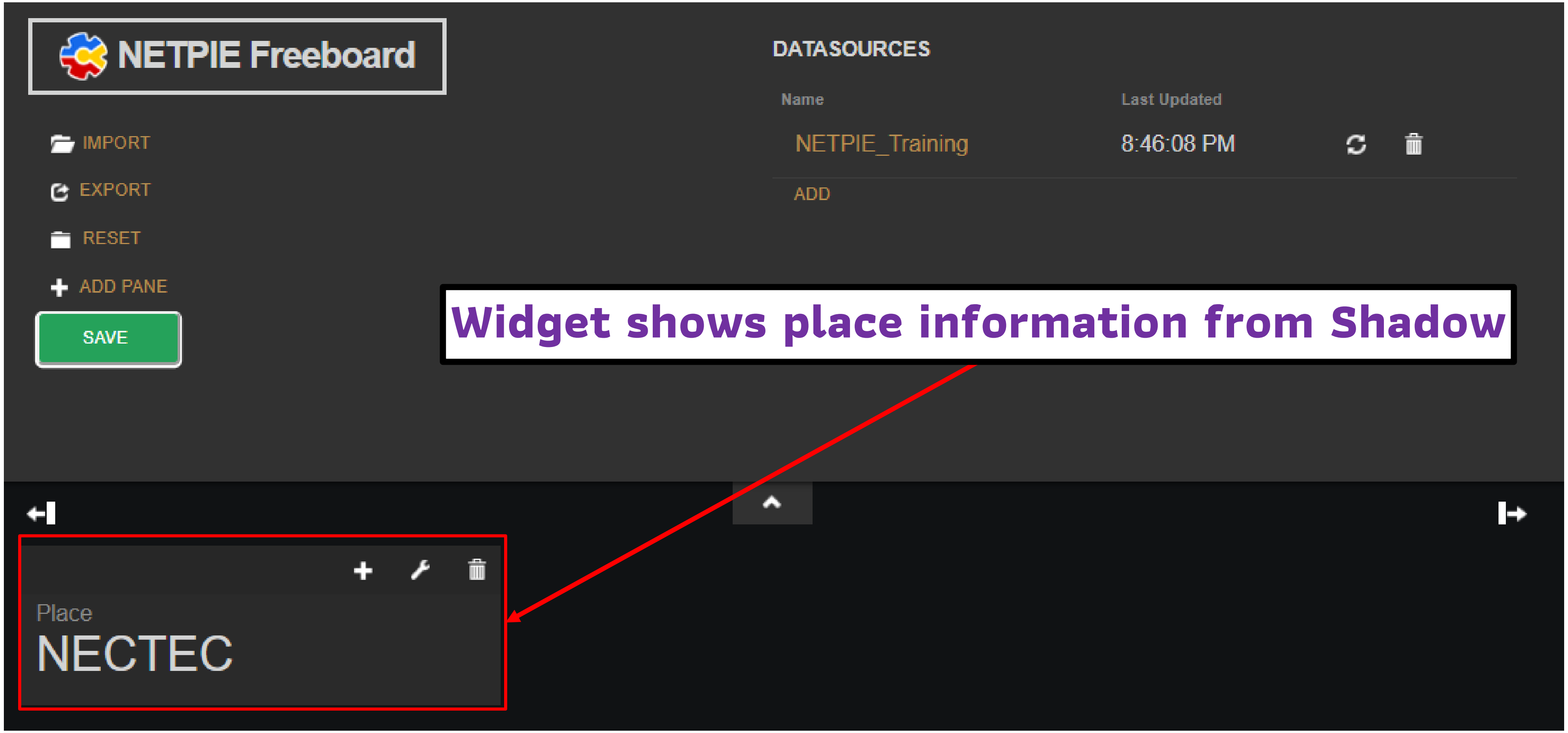
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



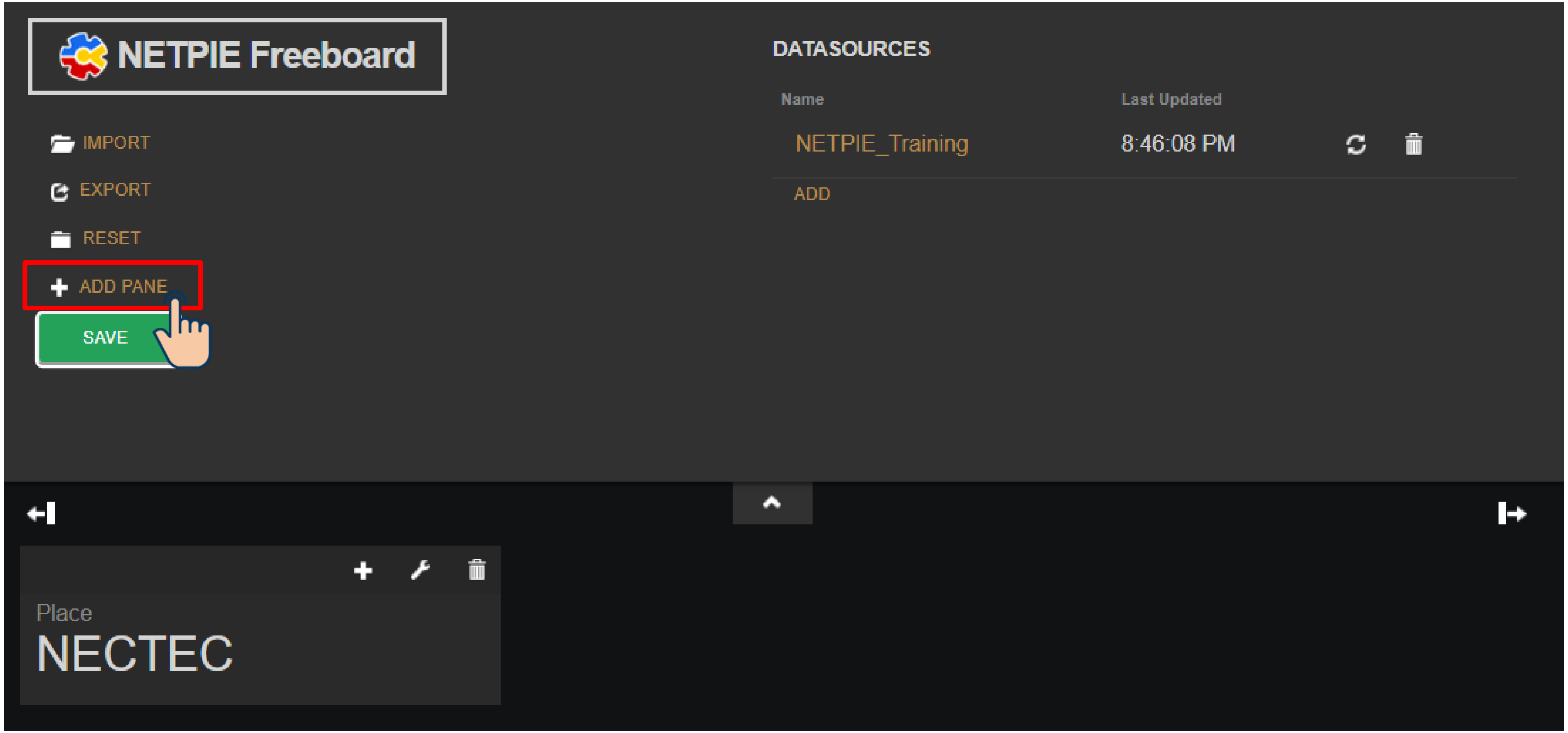
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



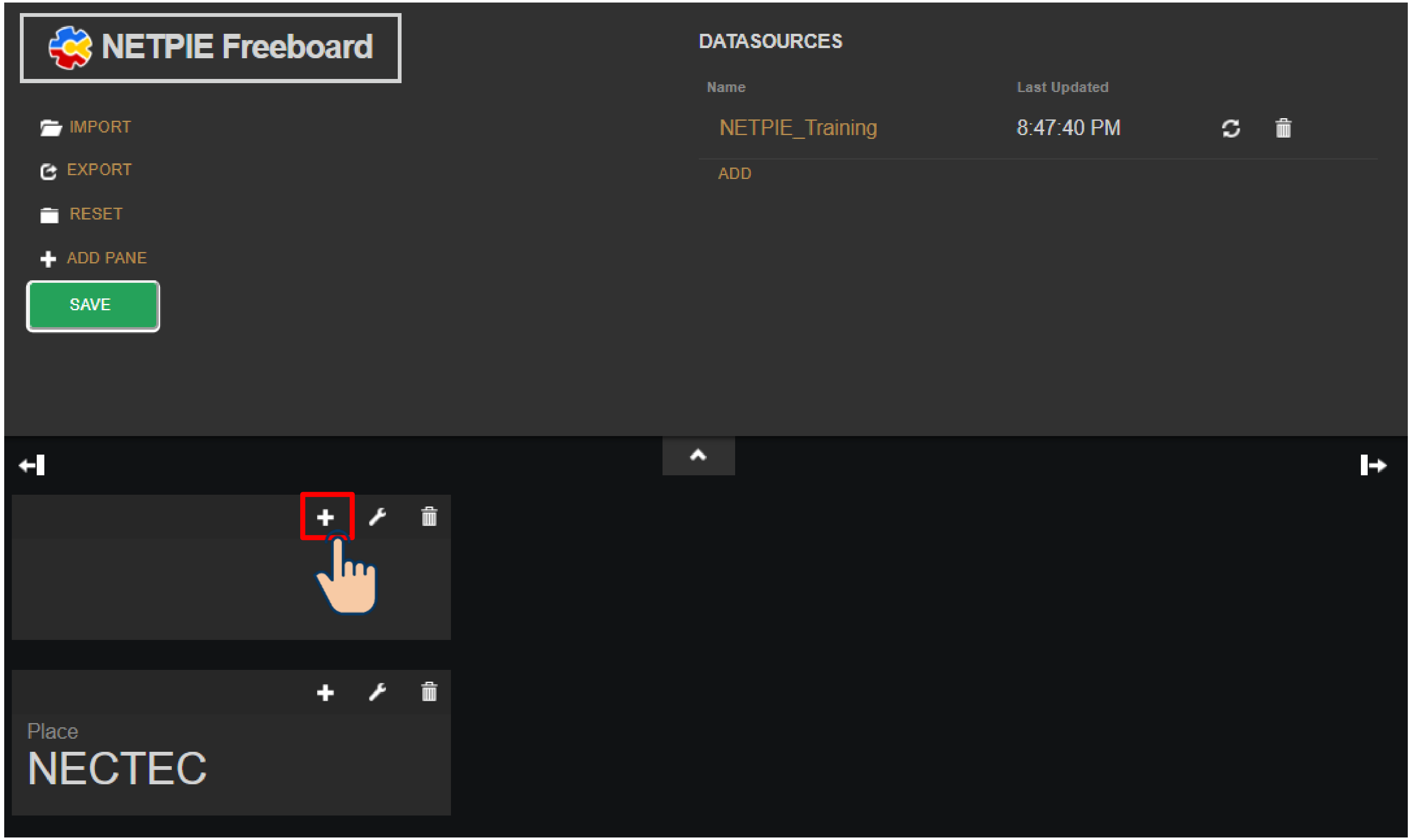
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



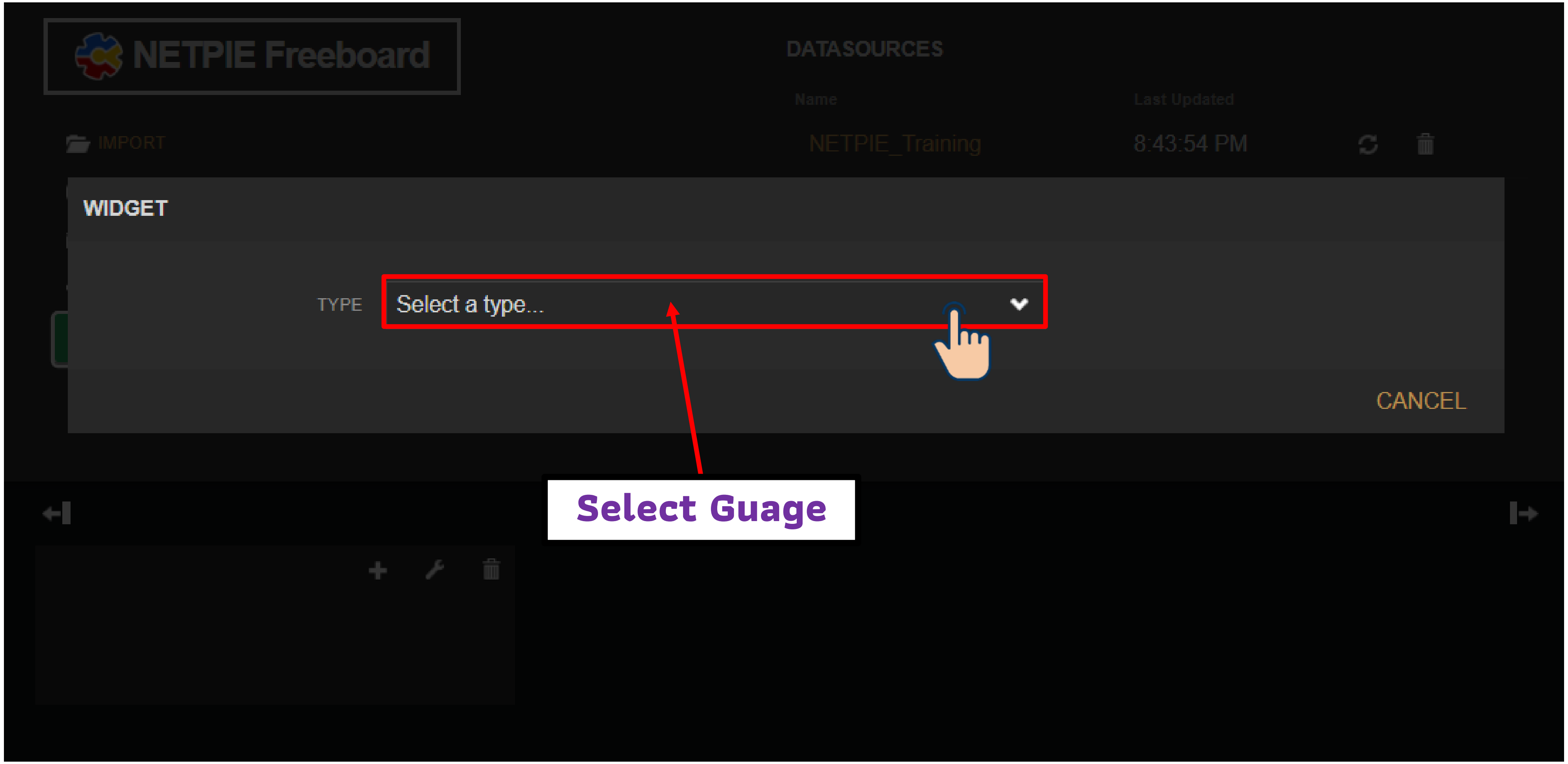
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



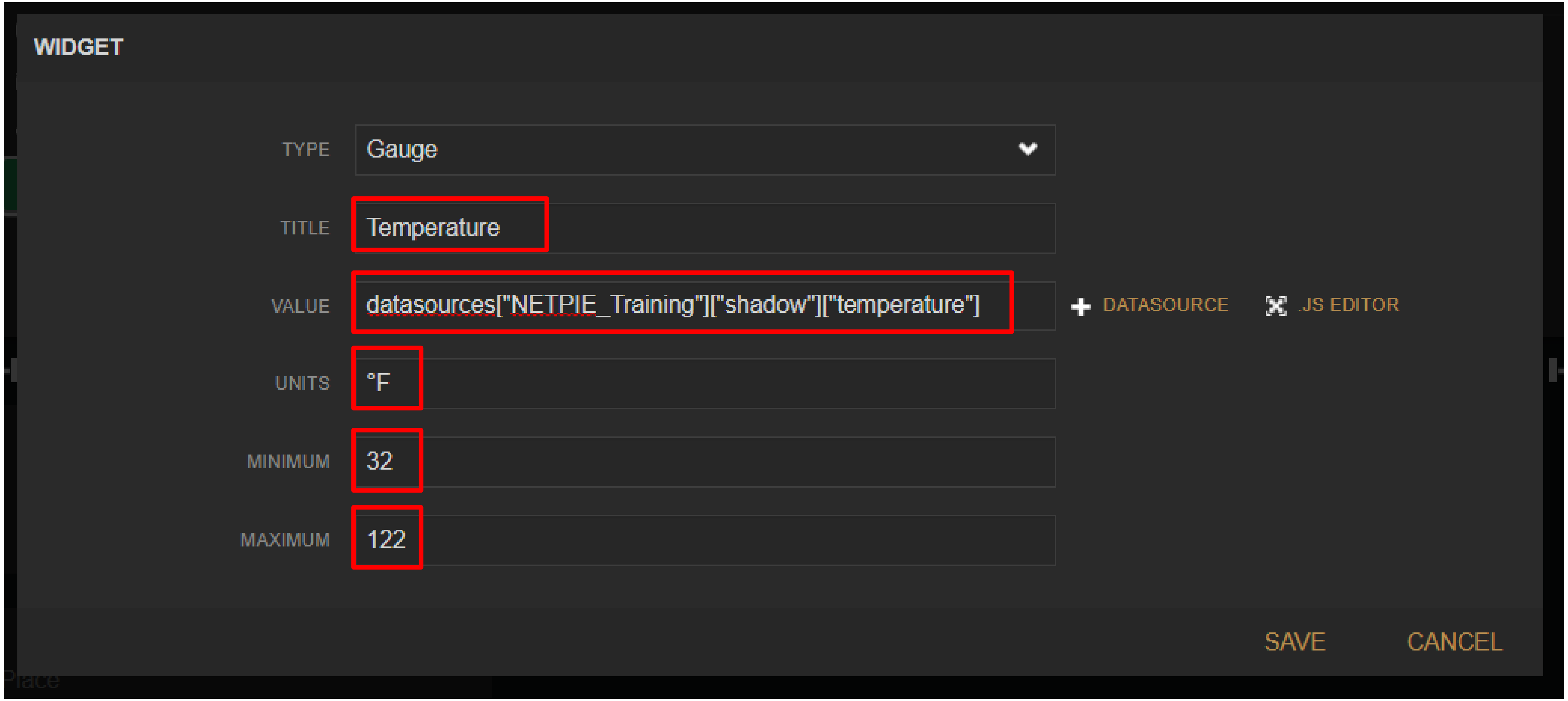
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



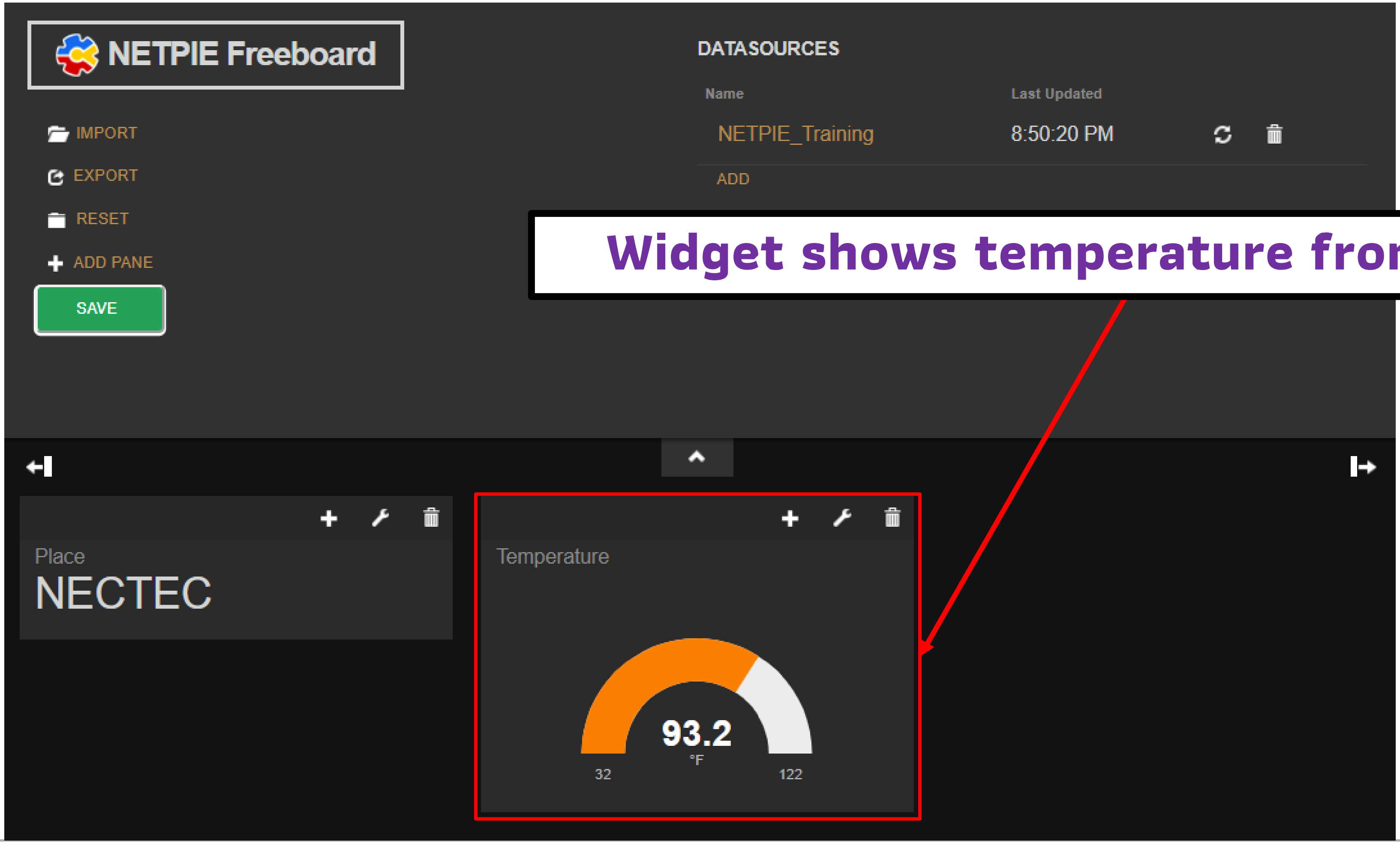
Exercise 6 : Construct a Freeboard on NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



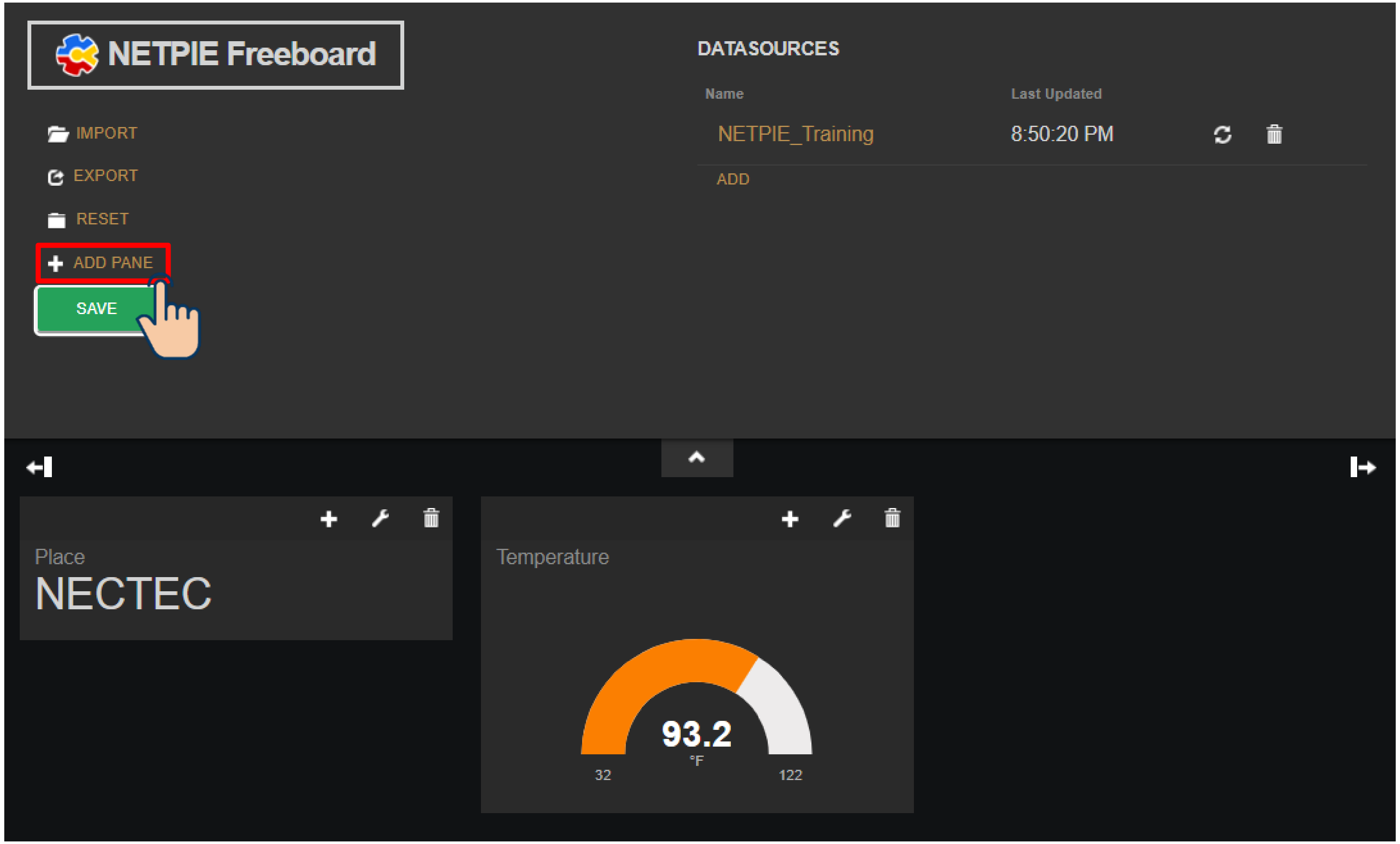
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



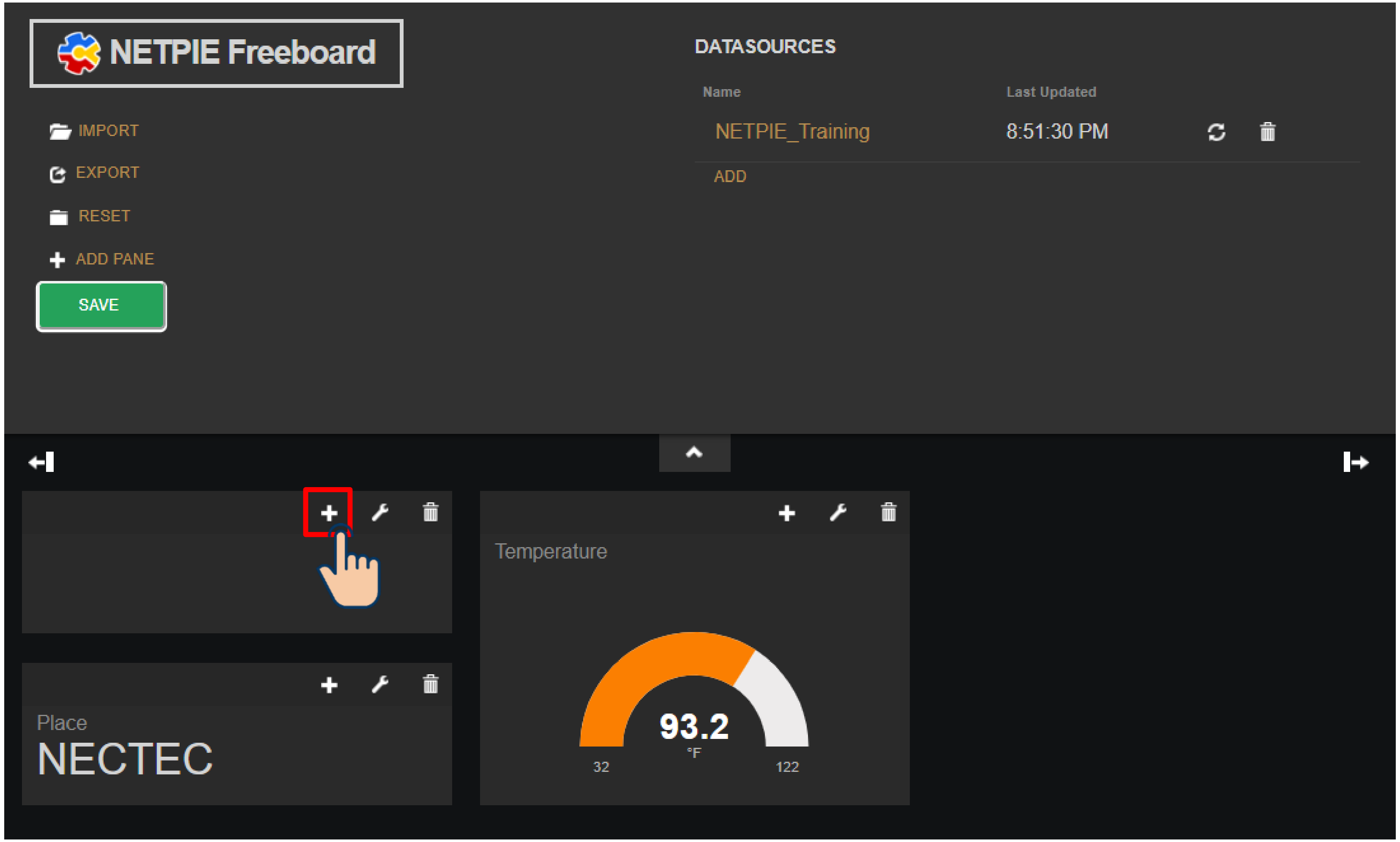
Widget shows temperature from Shadow

Exercise 6 : Construct a Freeboard on NETPIE2020



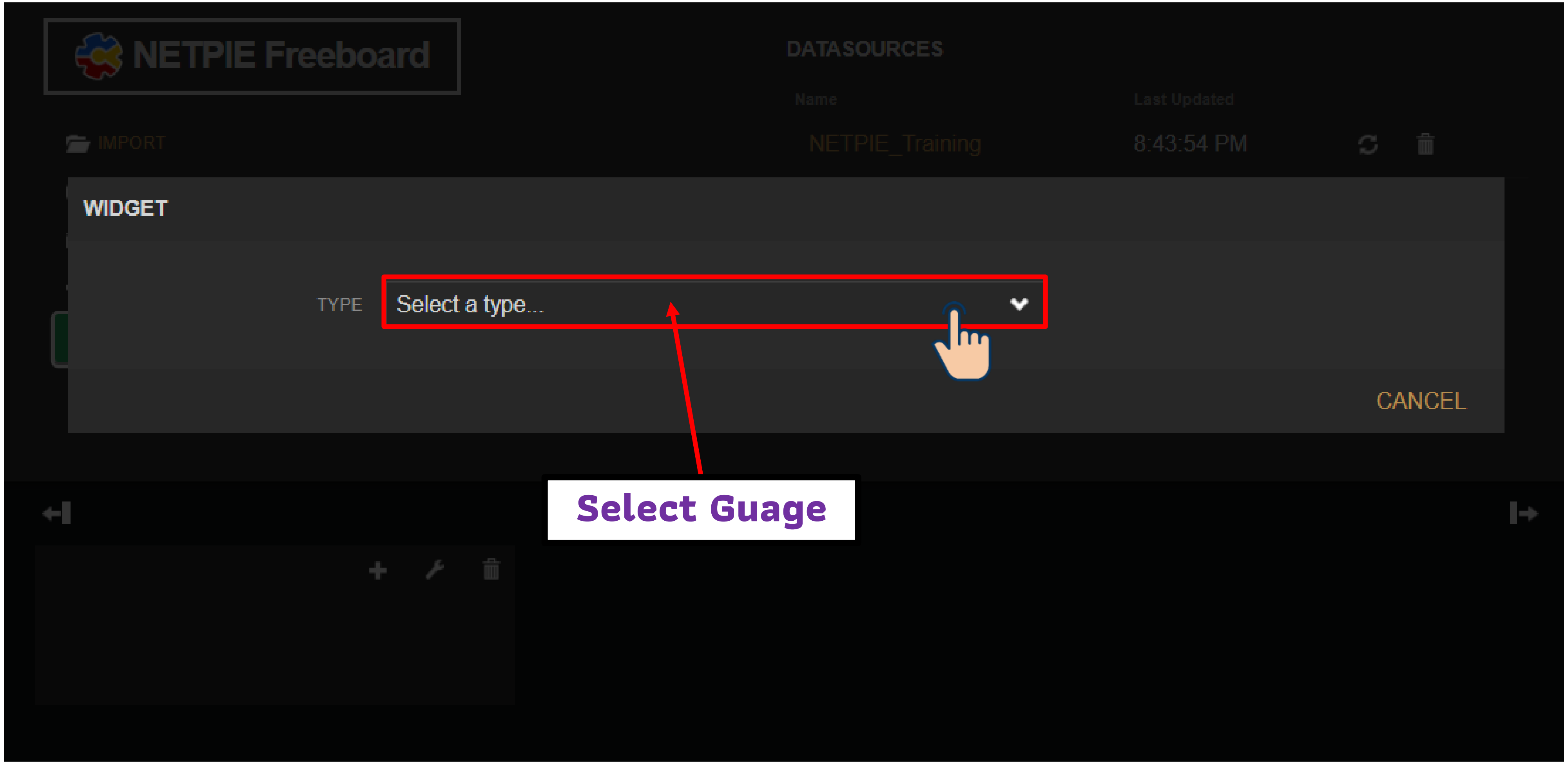
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

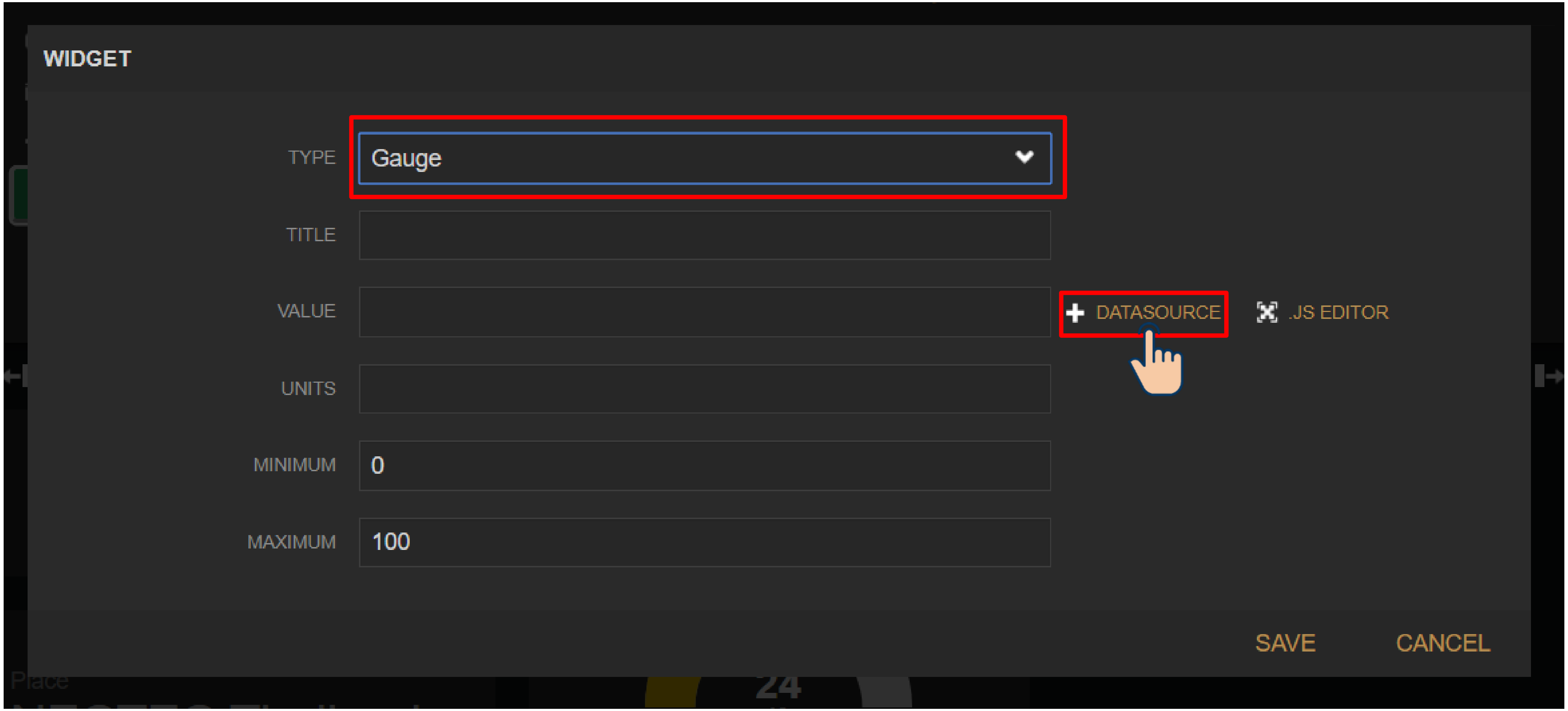


5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

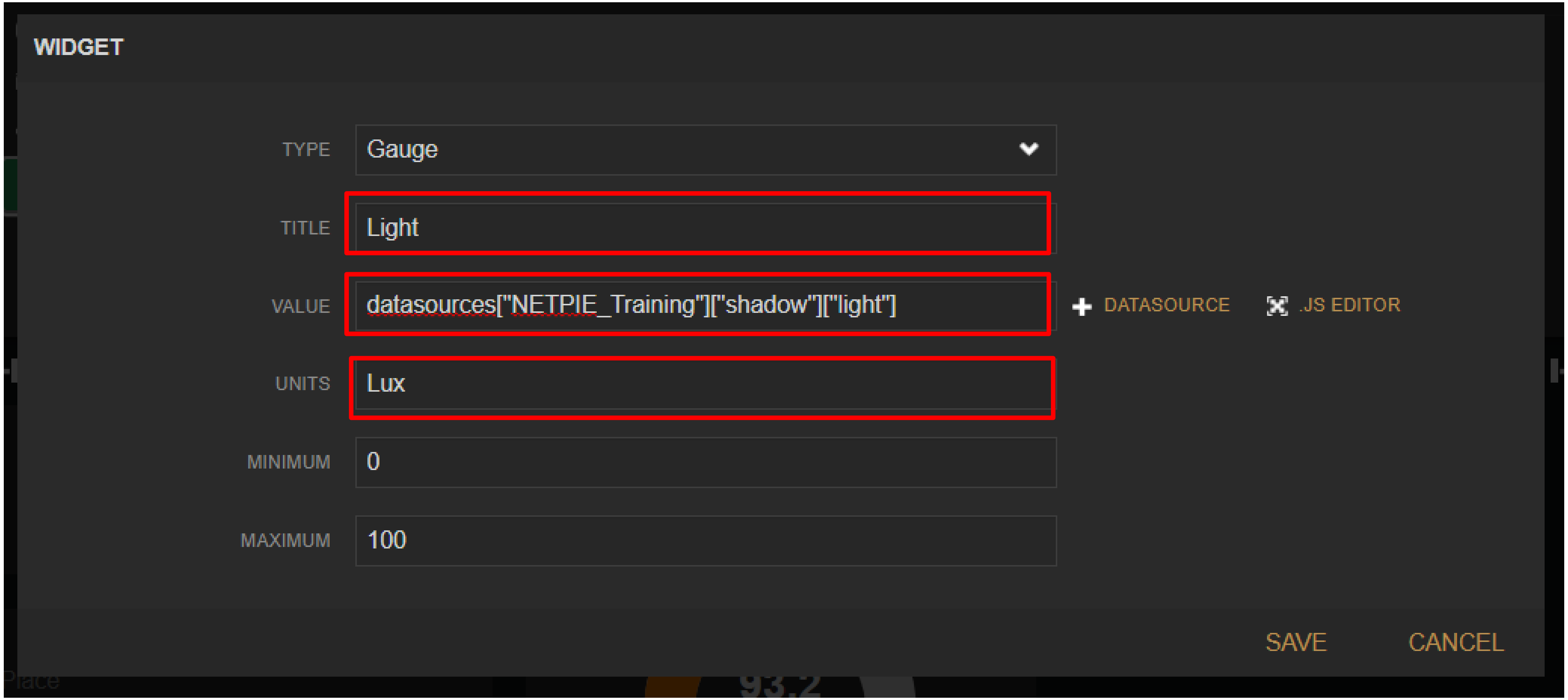


Exercise 6 : Construct a Freeboard on NETPIE2020

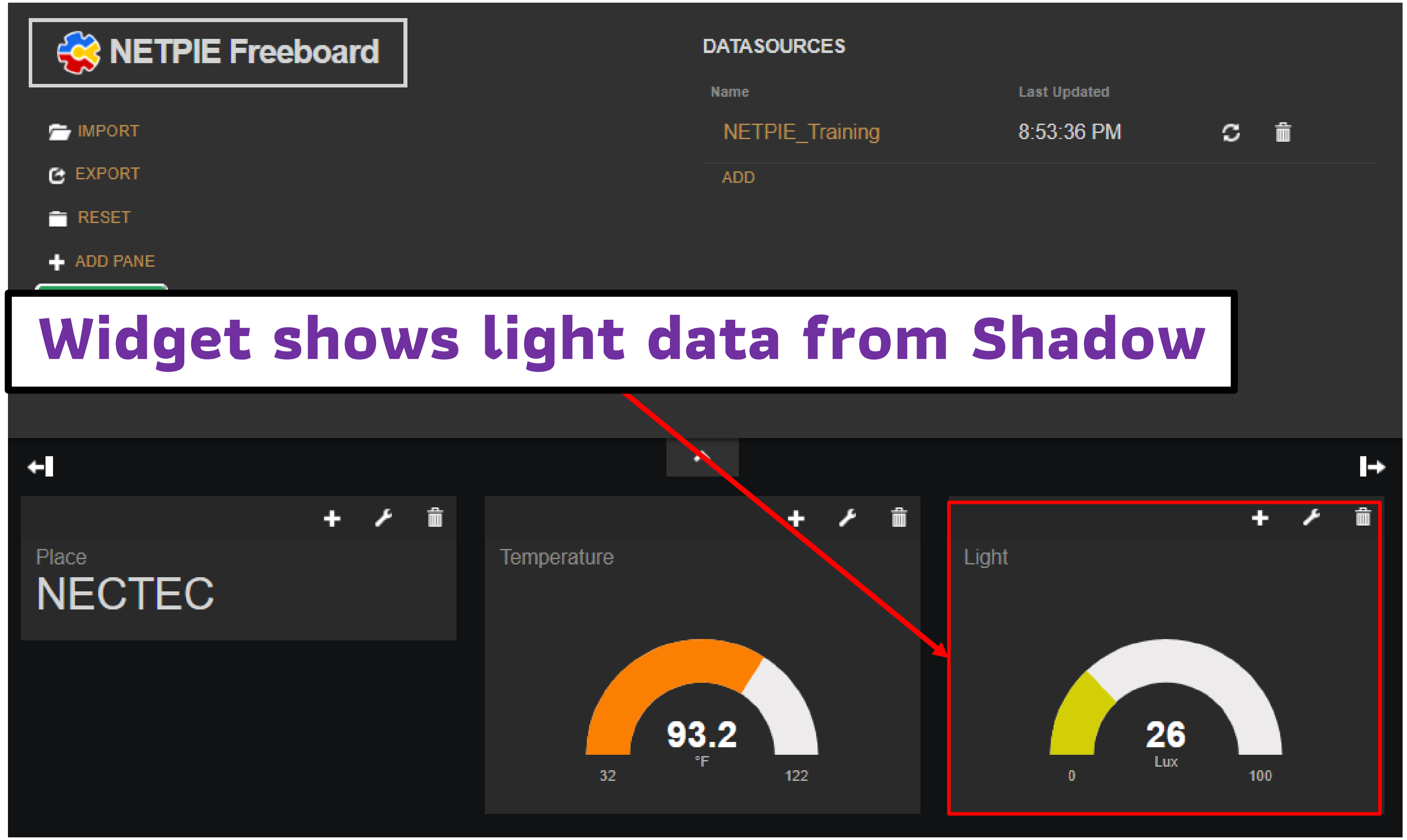


5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

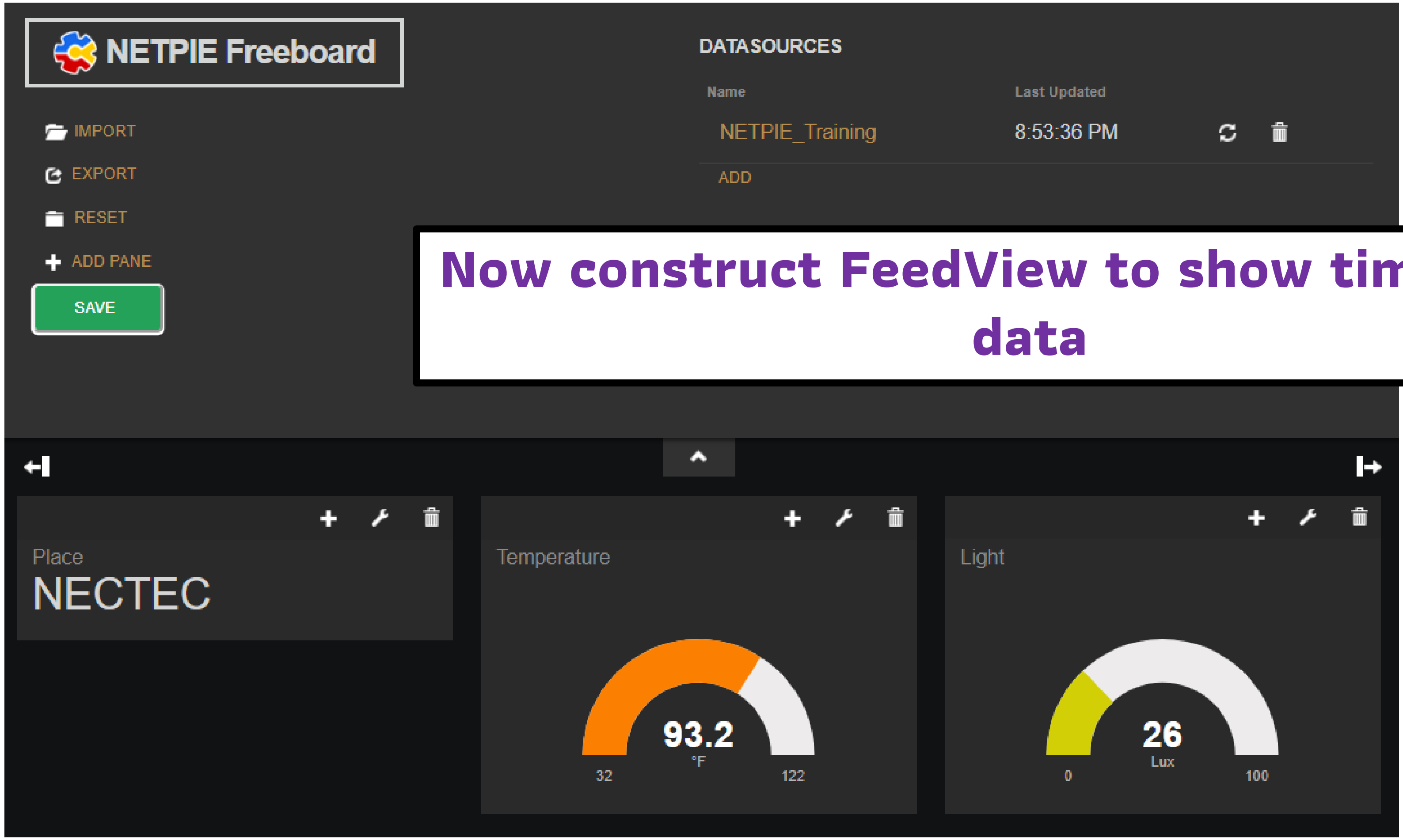


Exercise 6 : Construct a Freeboard on NETPIE2020



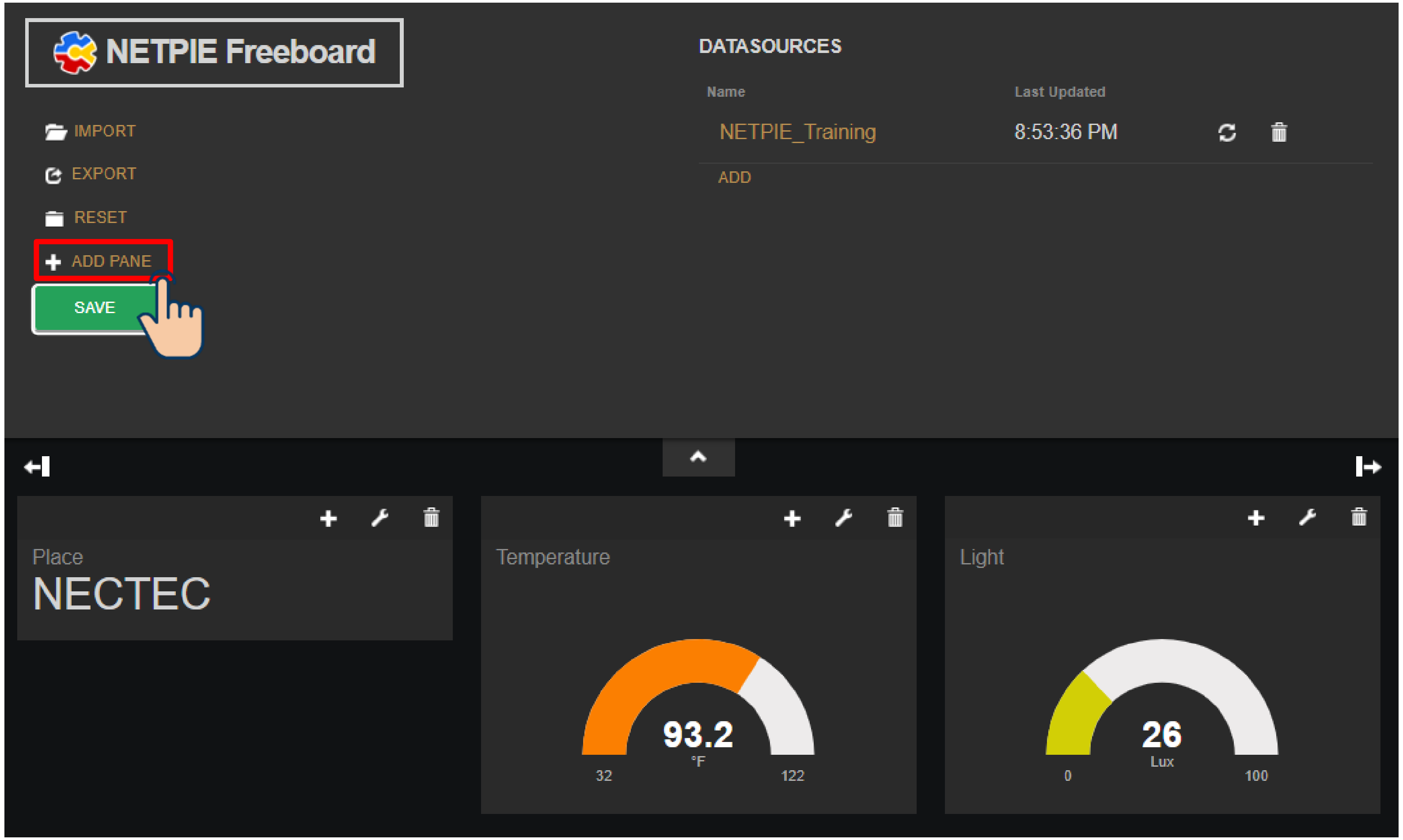
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



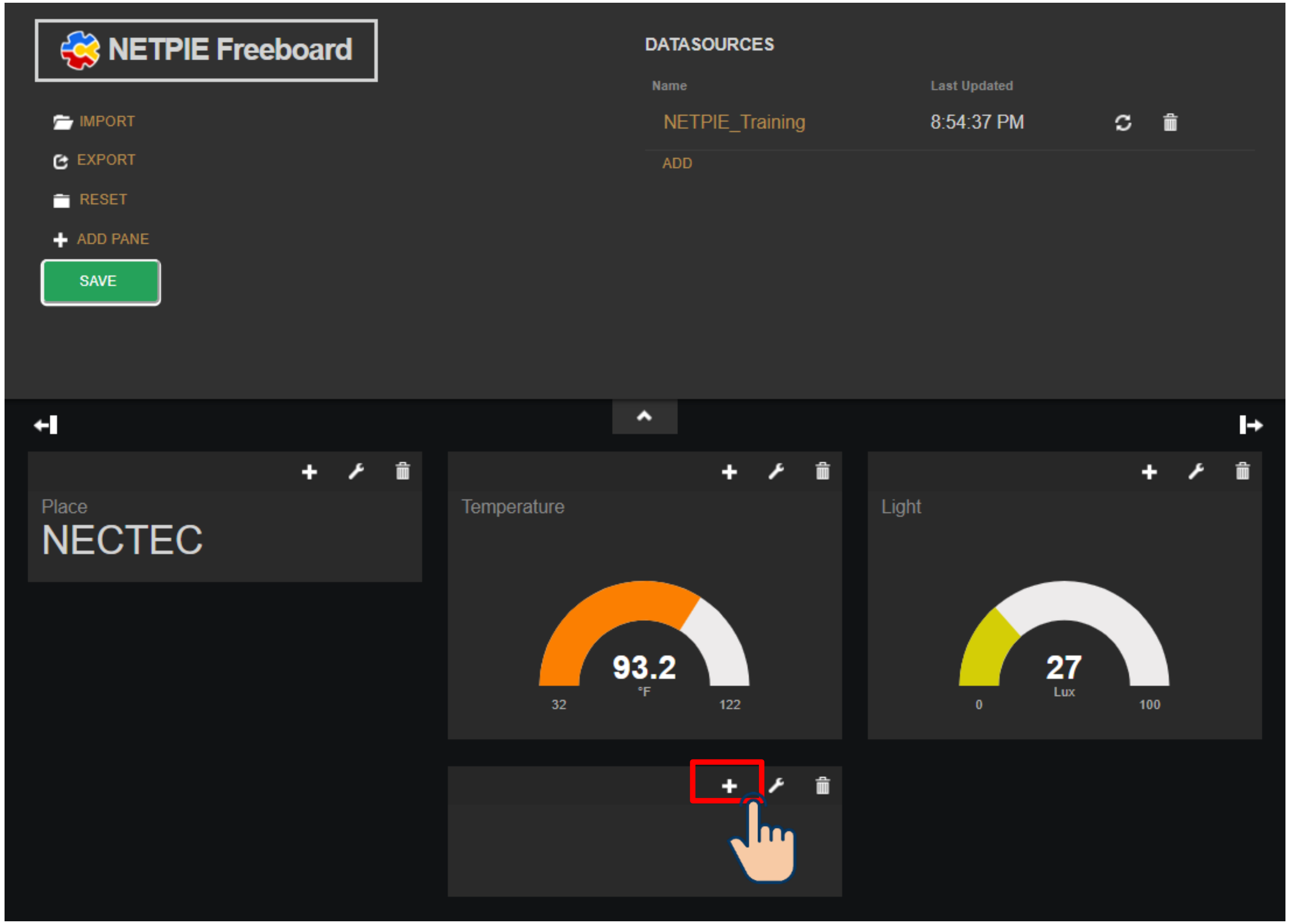
Now construct FeedView to show time series data

Exercise 6 : Construct a Freeboard on NETPIE2020



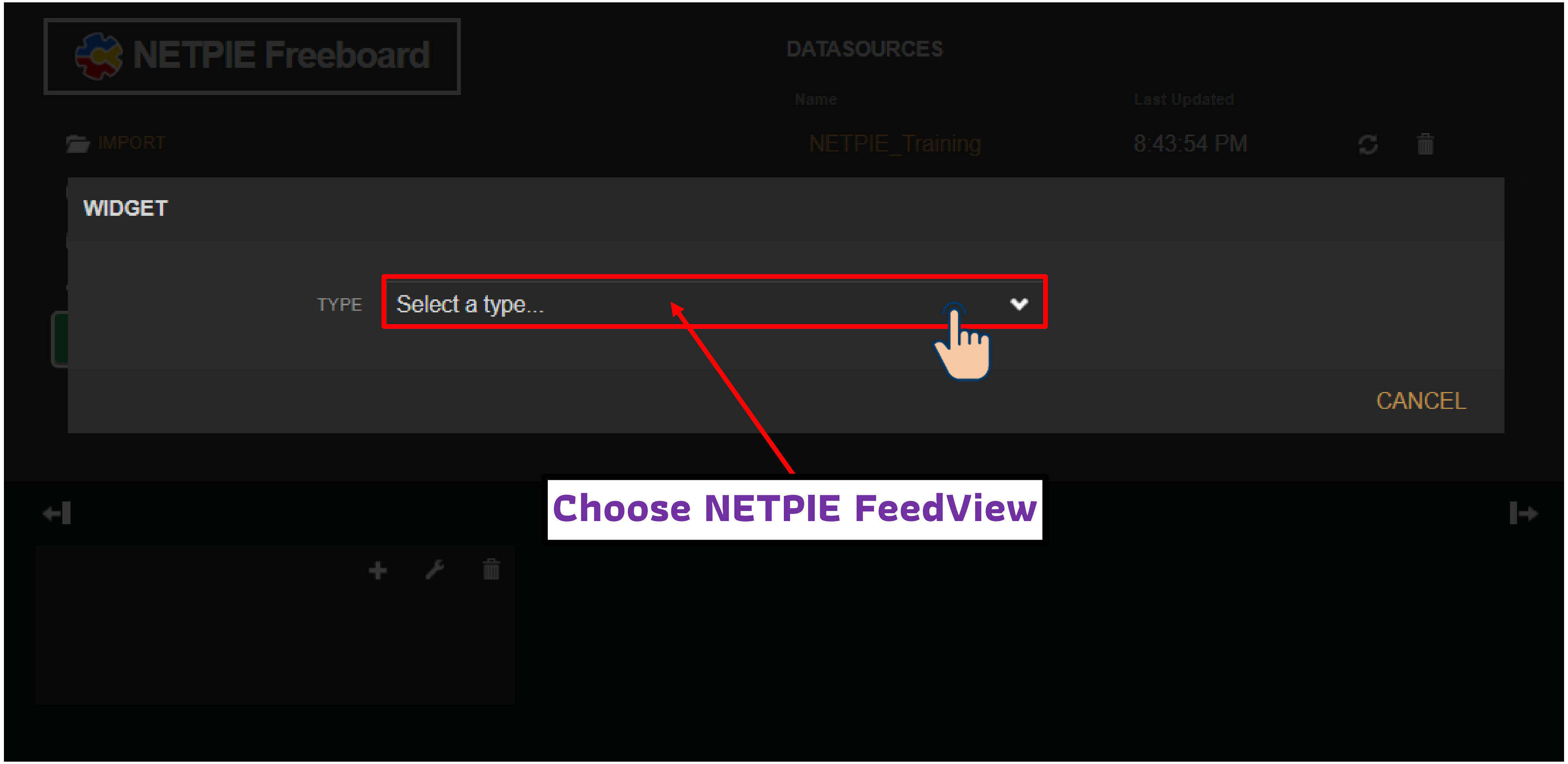
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



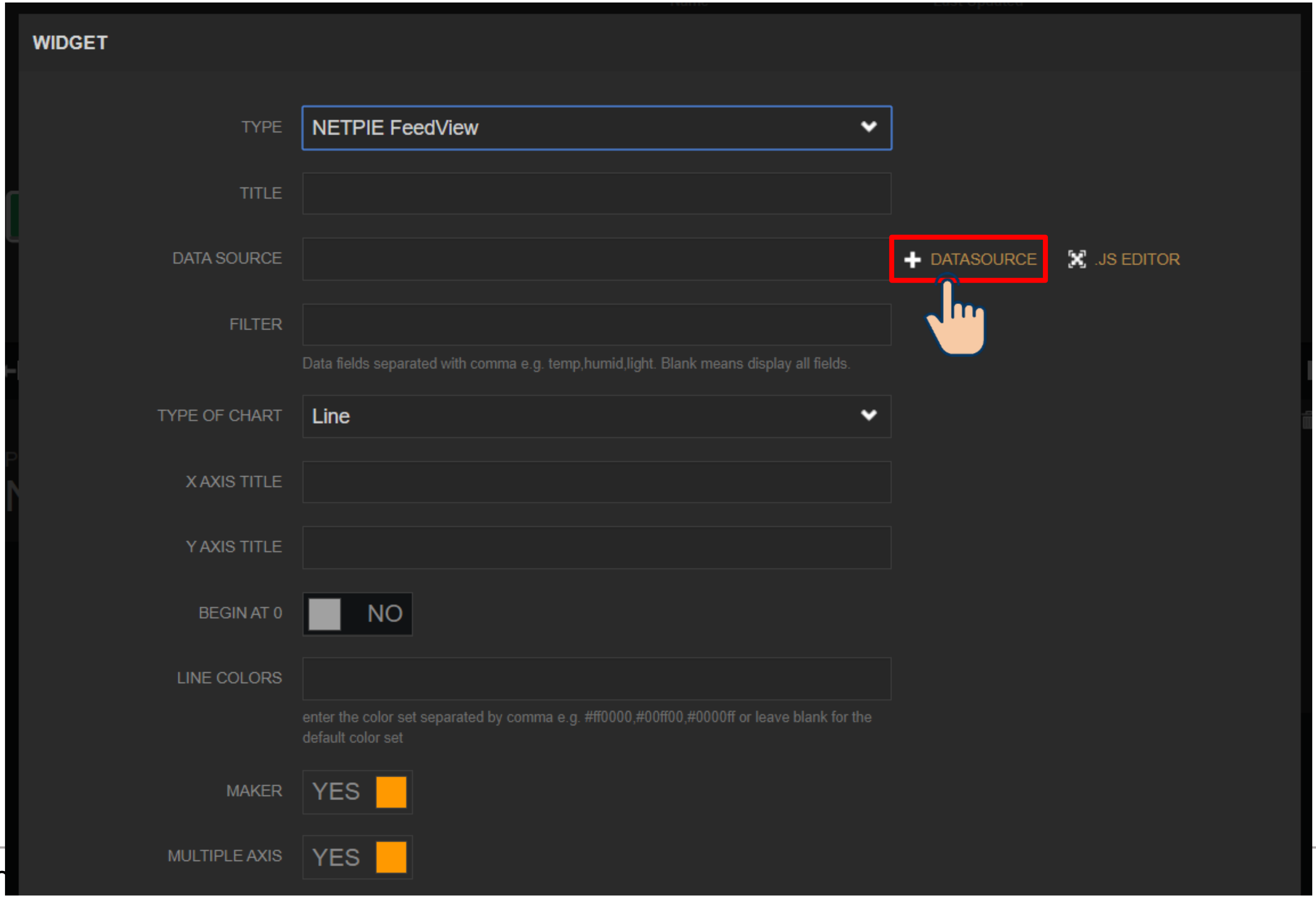
5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

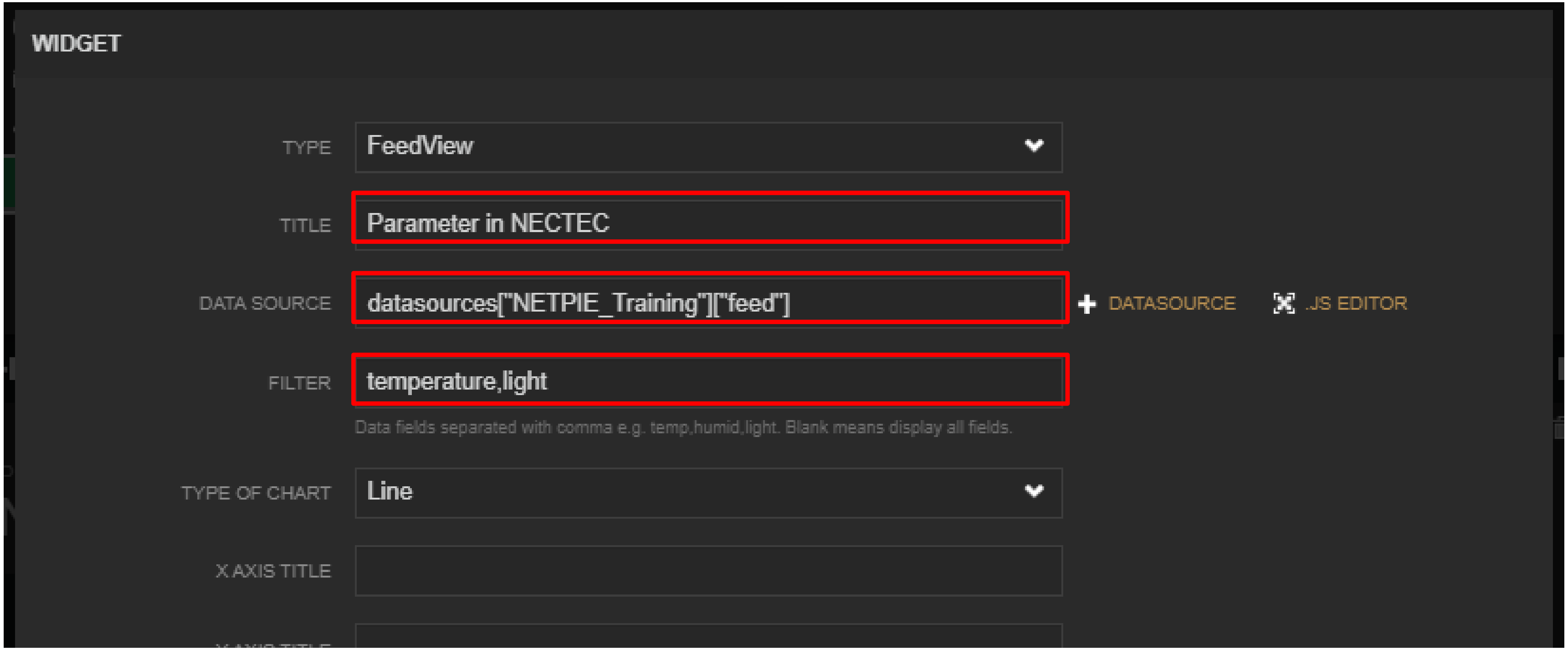


5 - Freeboard in NETPIE2020

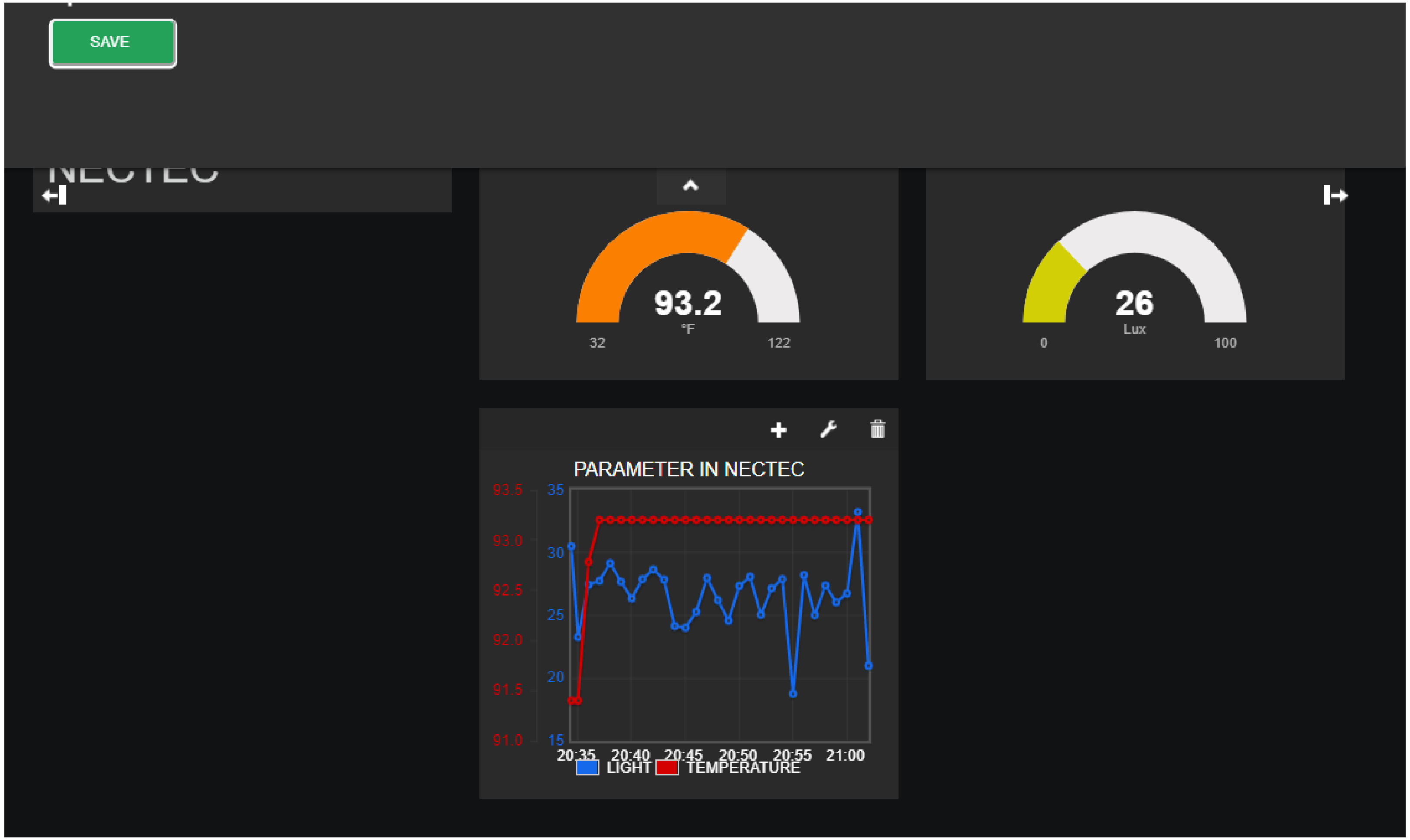
Exercise 6 : Construct a Freeboard on NETPIE2020



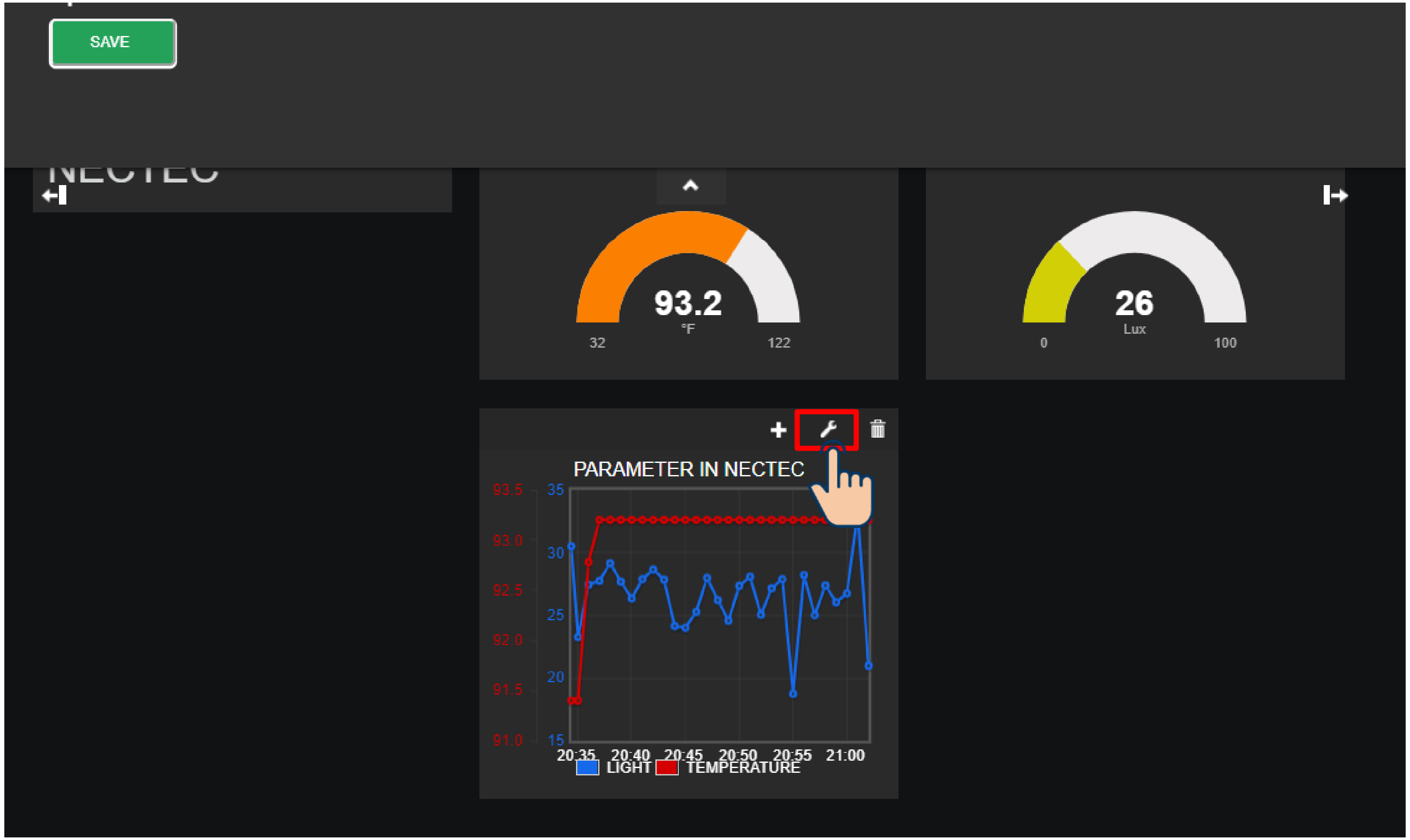
Exercise 6 : Construct a Freeboard on NETPIE2020



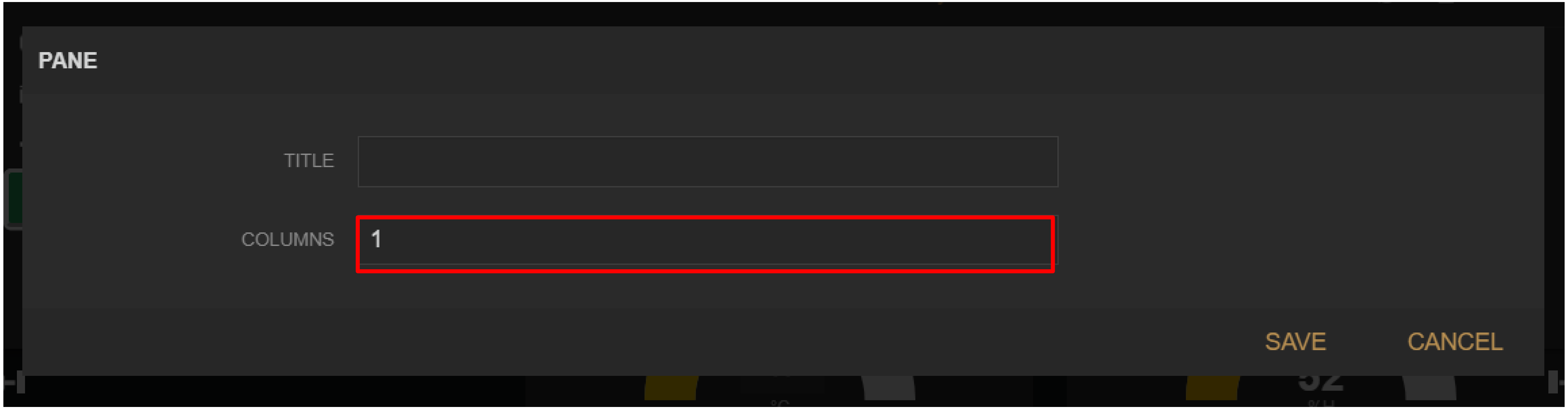
Exercise 6 : Construct a Freeboard on NETPIE2020



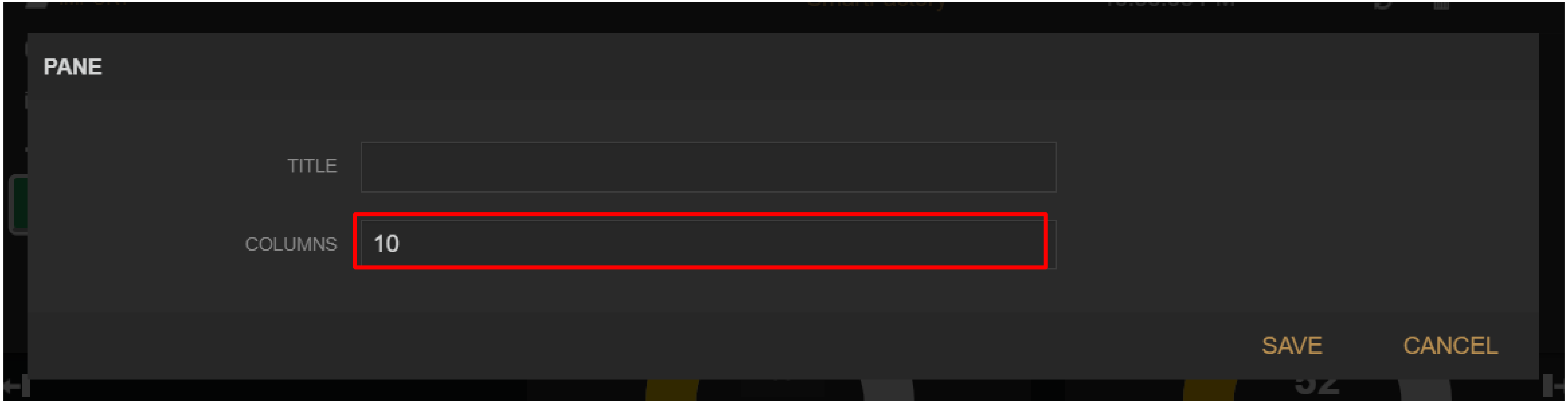
Exercise 6 : Construct a Freeboard on NETPIE2020



Exercise 6 : Construct a Freeboard on NETPIE2020

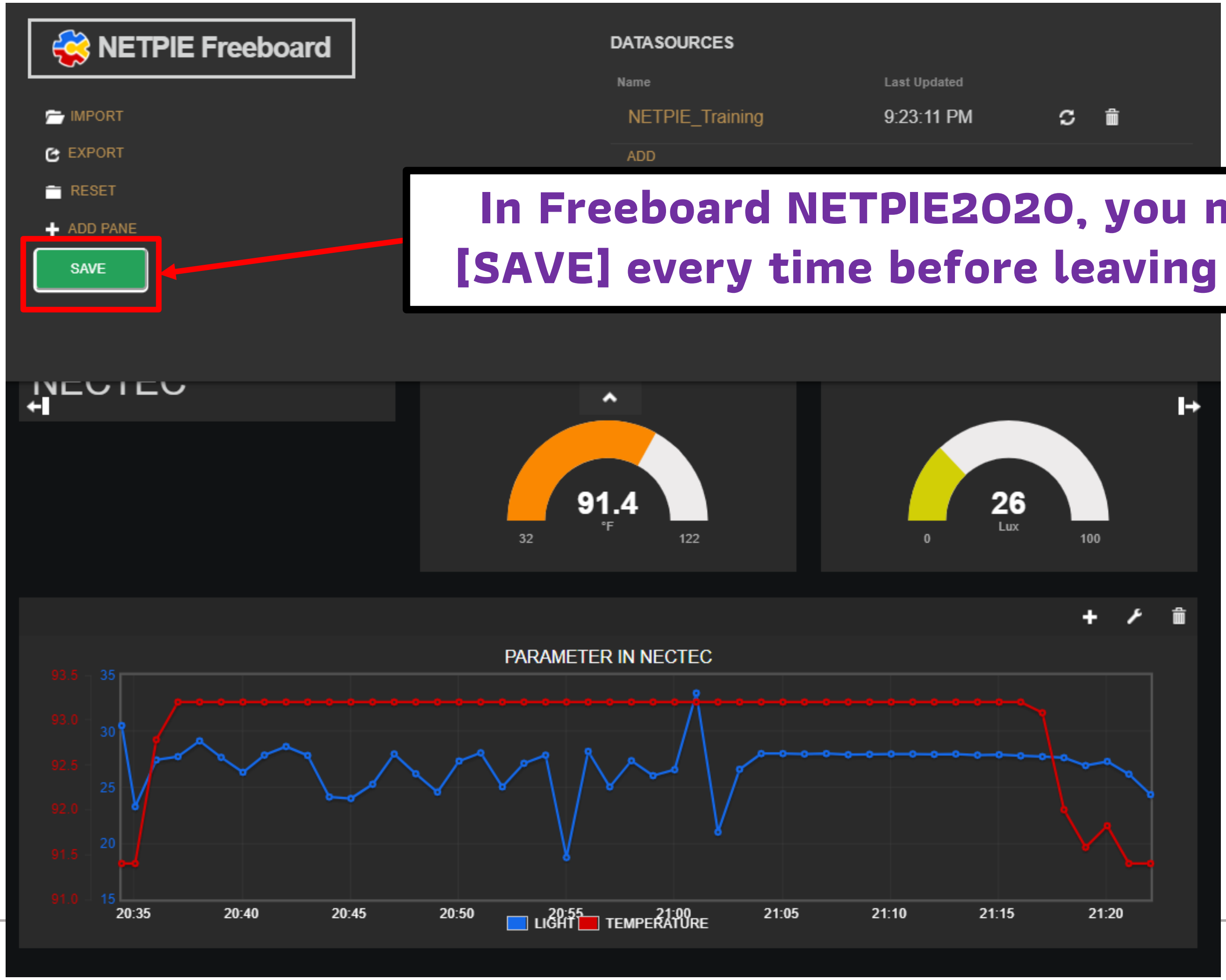


Exercise 6 : Construct a Freeboard on NETPIE2020



5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020



In Freeboard NETPIE2020, you must click [SAVE] every time before leaving Freeboard

5 - Freeboard in NETPIE2020

Exercise 6 : Construct a Freeboard on NETPIE2020

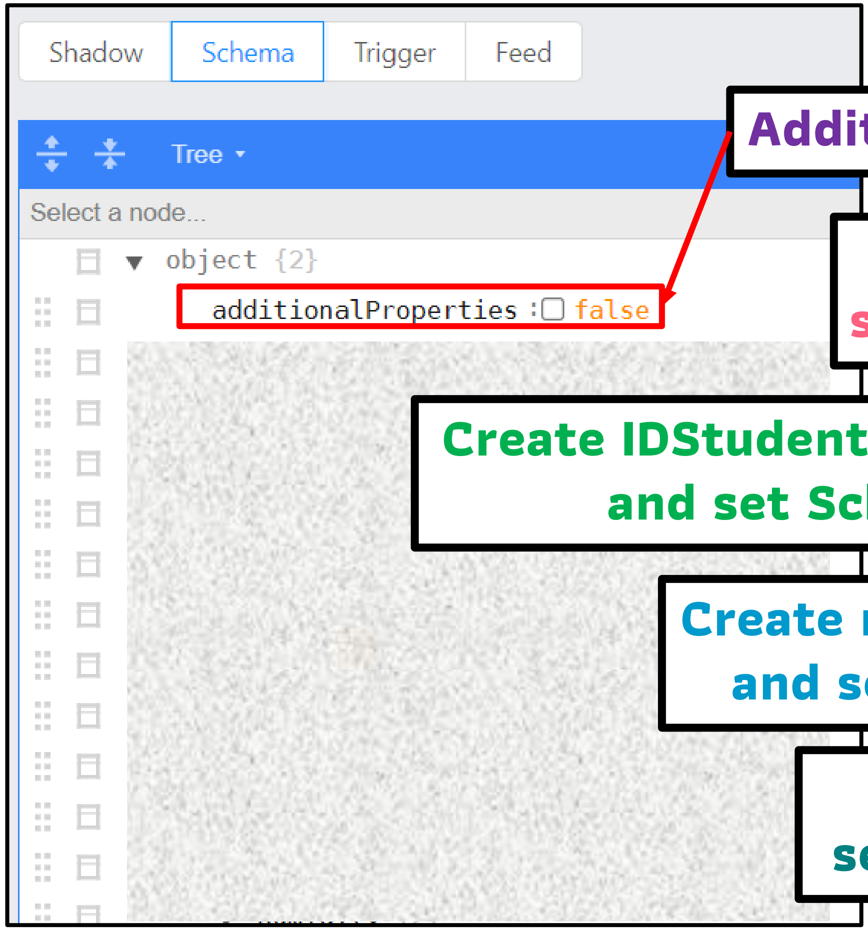
6 Workshop

NETPIE

where things chat

6 - Workshop

Create Shadow & Freeboard follow by my Image



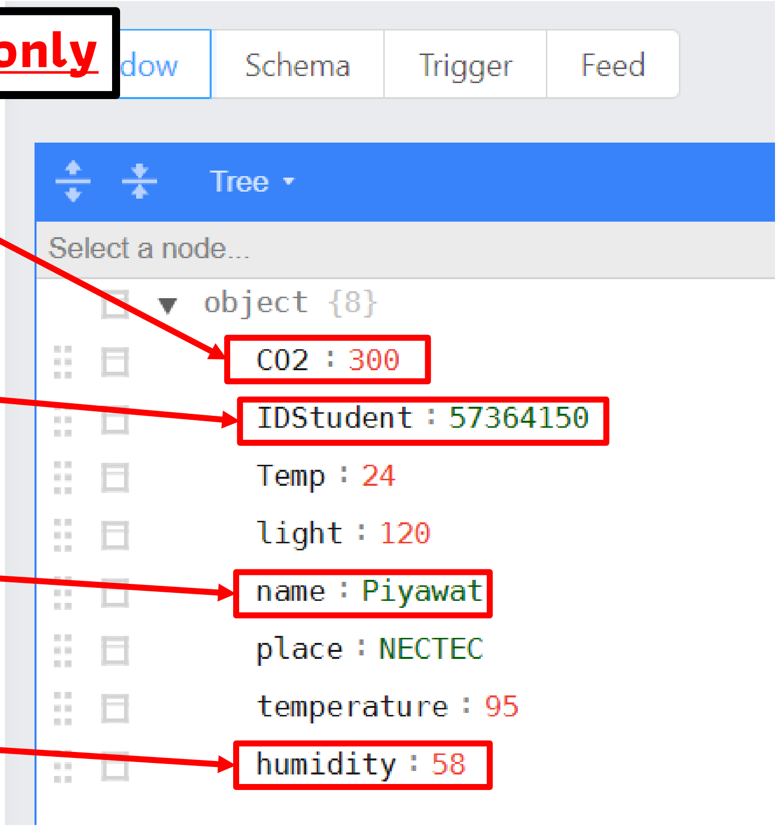
AdditionalProperties status **false only**

Create CO2 and set Schema [number, 7day]

Create IDStudent and input your IDStudent and set Schema [string, 7day]

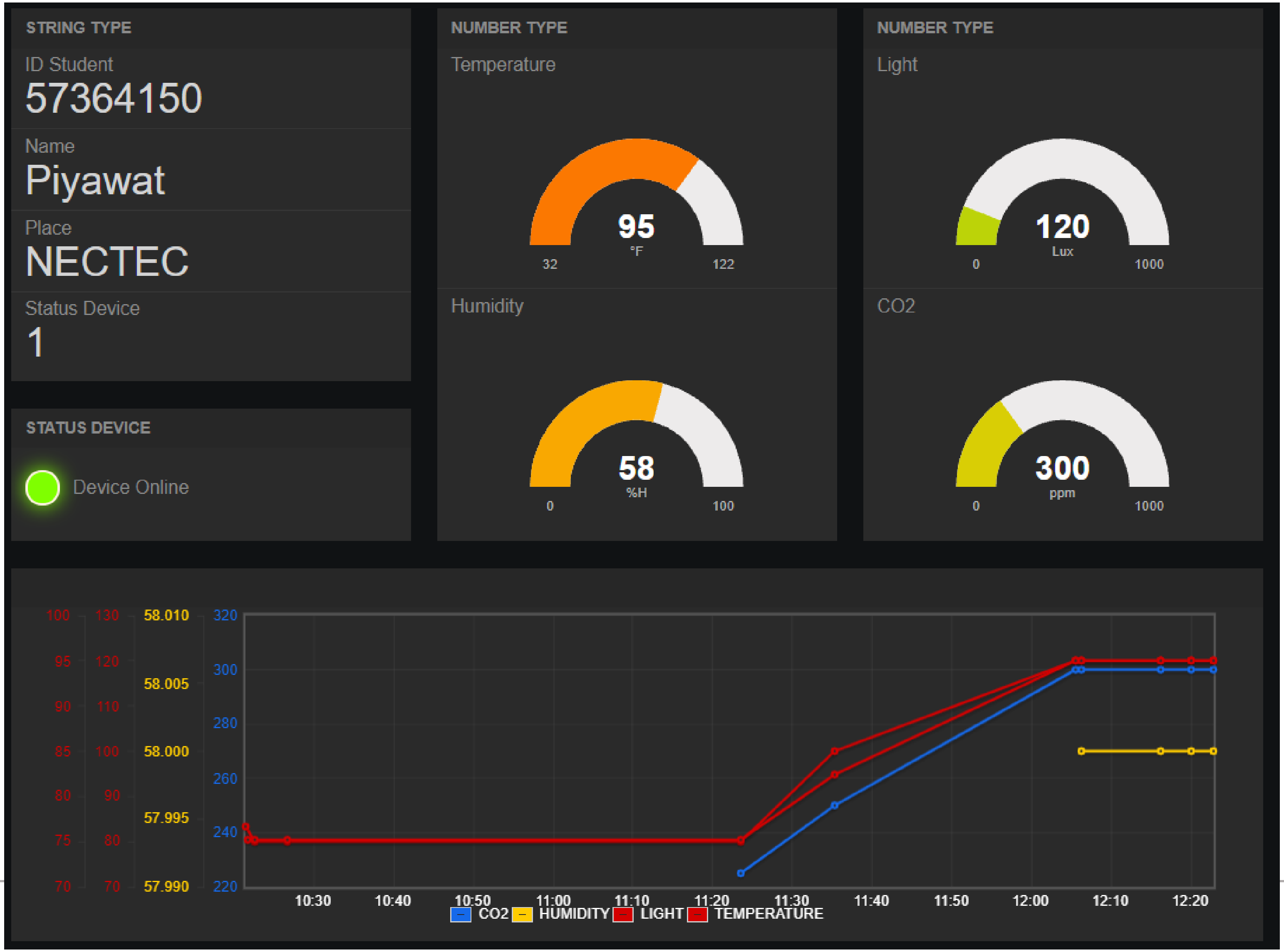
Create name and input your name and set Schema [string, 7day]

Create humidity and set Schema [number, 7day]



6 - Workshop

Create Shadow & Freeboard follow by my Image



6 - Workshop

How to send workshop???

1. Copy your device schema and paste on text file (.txt)
2. Capture your Freeboard Screen (.img or .png)
3. Drop 2 File in Google Drive

Drop your workshop at

Link --> https://bit.ly/3m4isJj

Set File Name is W follow by your IDStudent

Such as W1_57364150, W2_57364150

(W1 = Schema File, W2 = Freeboard File)

Internet of Things & NETPIE2020 – Workshop

(For 11 Sep'21)

Mr. Piyawat Jomsathan (Tae)
Cyber-Physical Systems (CPS)
National Electronics and Computer Technology Center (NECTEC), Thailand

สวทช NSTDA 30th Anniversary
NECTEC a member of NSTDA

Download Document & Software

Link --> https://bit.ly/3BXJzeh

Main Topic

1 Data Management in NETPIE2020 (Trigger & Event Hook)

2 MQTT Library in Arduino IDE (Pubsubclient)

3 Freeboard in NETPIE2020 (Control Widget)

4 RESTful API

5 Workshop

1

Data Management in NETPIE2020 (Trigger & Event Hook)

NETPIE
where things chat

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Device Trigger and Event Hook

It is a system that binds to changing Device Shadow information to external actions (Event Hook), such as setting alerts according to different status. According to the condition of the device set by the Trigger will be declared in JSON format:

```
{
    "enabled": true,
    "trigger": [
        {
            "action": "EVENT_HOOK_NAME",
            "event": "SHADOW.UPDATED or DEVICE.STATUSCHANGED",
            "condition": "Operation List ==, !=, >, >=, <, <=, in",
            "msg": "text",
            "option": {}
        }
    ]
}
```

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Device Trigger and Event Hook

Trigger Format consists of 2 parts

1. enable is used to turn the Trigger on/off

```
{
   "enabled": true,
   "trigger": [
      {
         "action": "EVENT_HOOK_NAME",
         "event": "SHADOW.UPDATED or DEVICE.STATUSCHANGED",
         "condition": "Operation List ==, !=, >, >=, <, <=, in",
         "msg": "text",
         "option": {}
      }
   ]
}
```

Device Trigger and Event Hook

- action : what Trigger has to do when an event occurs. Specify Event Hook name.

Trigger Format consist of 2 parts

2. trigger is for Trigger settings

- event : corresponds to two types of change in Device Shadow
  SHADOW.UPDATED : happens when Device Shadow Data changes according to certain condition (in this case we have to specify some condition, otherwise the Trigger would not fire)
  DEVICE.STATUSCHANGED : happens when the Device changes its platform connection status from Online to Offline, or vice versa.

```
{
  "enabled": true,
  "trigger": [
    {
      "action": "EVENT_HOOK_NAME",
      "event": "SHADOW.UPDATED or DEVICE.STATUSCHANGED",
      "condition": "Operation List ==, !=, >, >=, <, <=, in",
      "msg": "text",
      "option": {}
    }
  ]
}
```

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Device Trigger and Event Hook

Trigger Format consist of 2 parts

```
{
  "enabled": true,
  "trigger": [
    {
      "action": "EVENT_HOOK_NAME",
      "event": "SHADOW.UPDATED or DEVICE.STATUSCHANGED",
      "condition": "Operation List ==, !=, >, >=, <, <=, in",
      "msg": "text",
      "option": {}
    }
  ]
}
```

Condition : conditional change in Device Shadow Data . Use in case of SHADOW.UPDATED

msg : a message to notify the user when Trigger fires

option : use for specify some other parameters

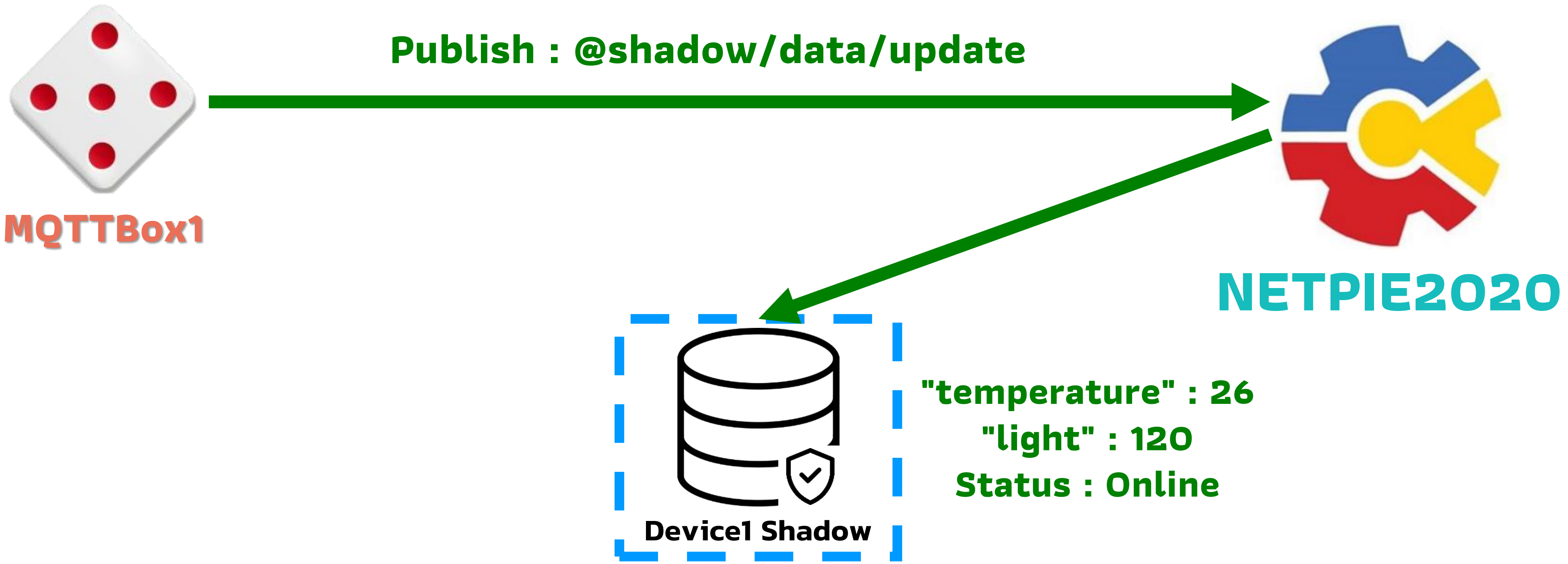
For reference, variables in Trigger can be found in
https://docs.nexpie.io/device-config.html#device-trigger-and-event-hook

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Device Trigger and Event Hook

Trigger and Event Hook example

Event Hooks
LINE NOTIFY

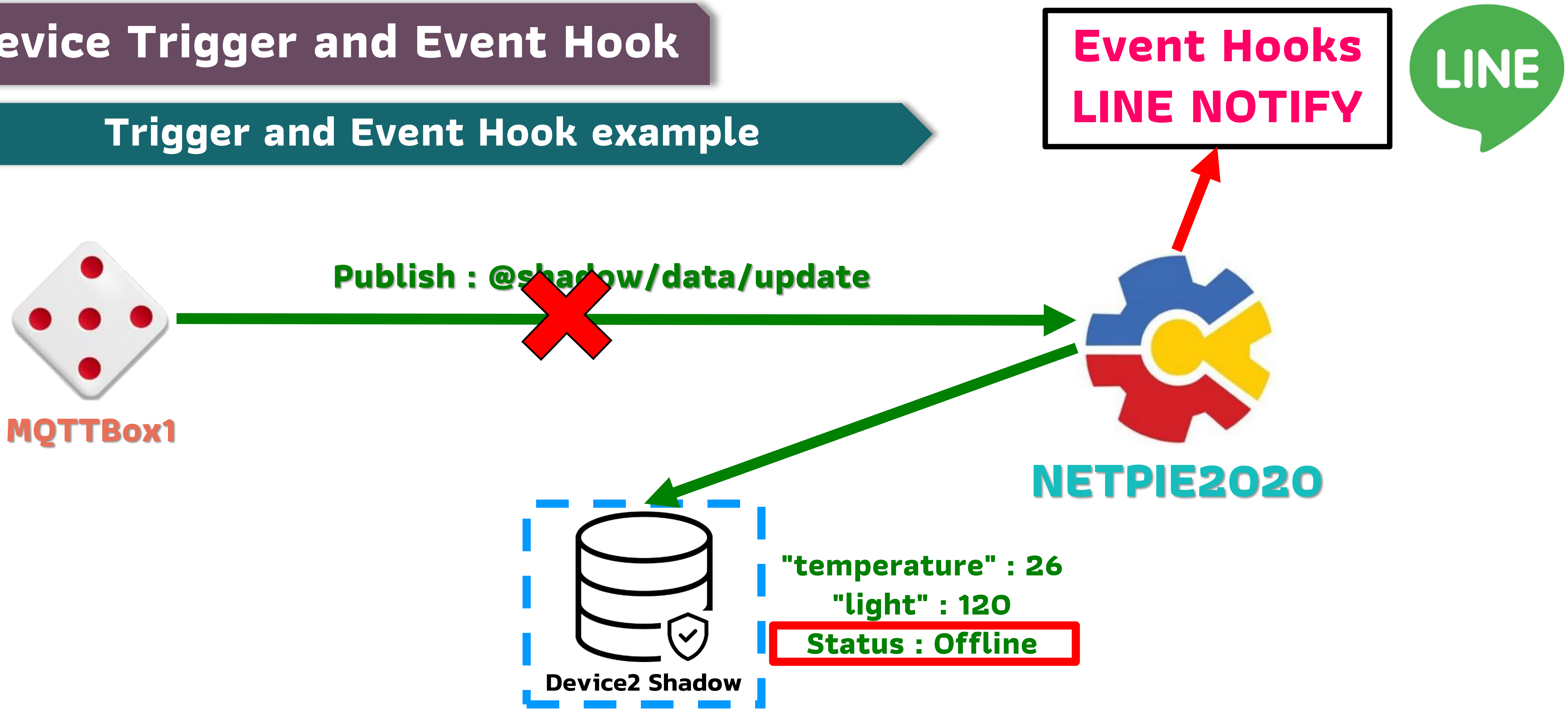


Trigger : DEVICE.STATUSCHANGE

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Device Trigger and Event Hook

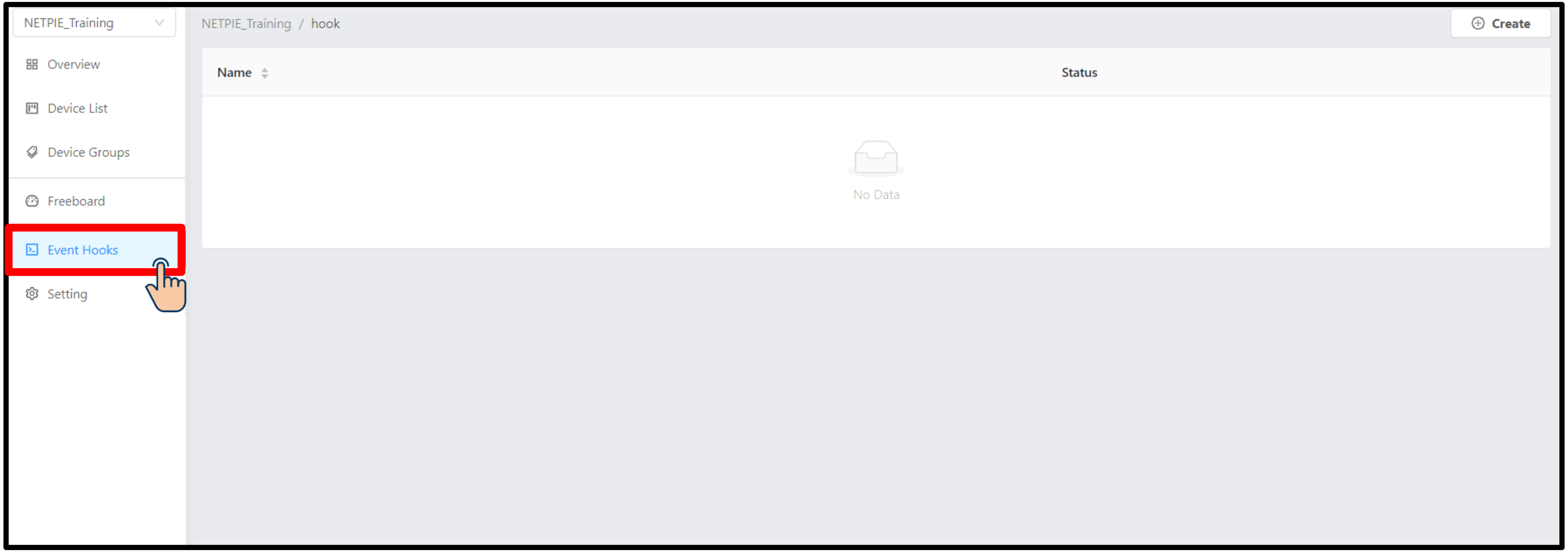
Trigger and Event Hook example



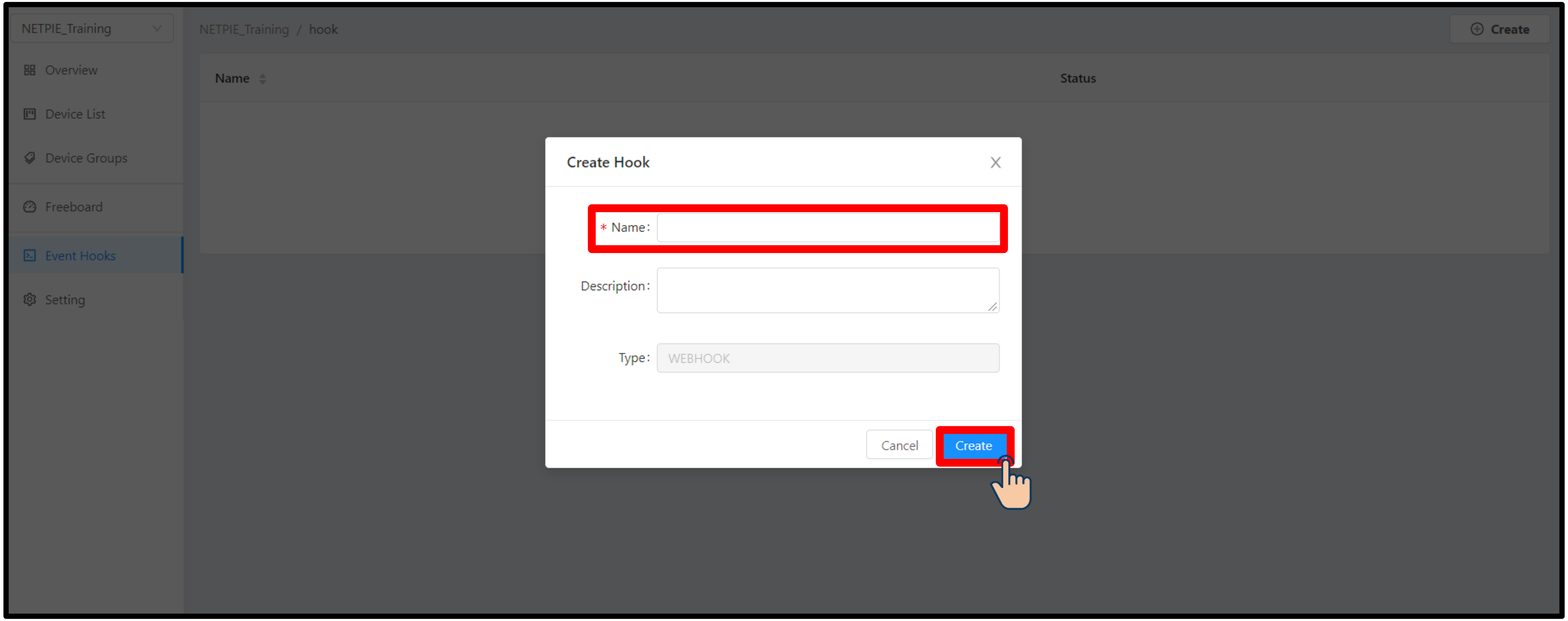
Trigger : DEVICE.STATUSCHANGE

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook

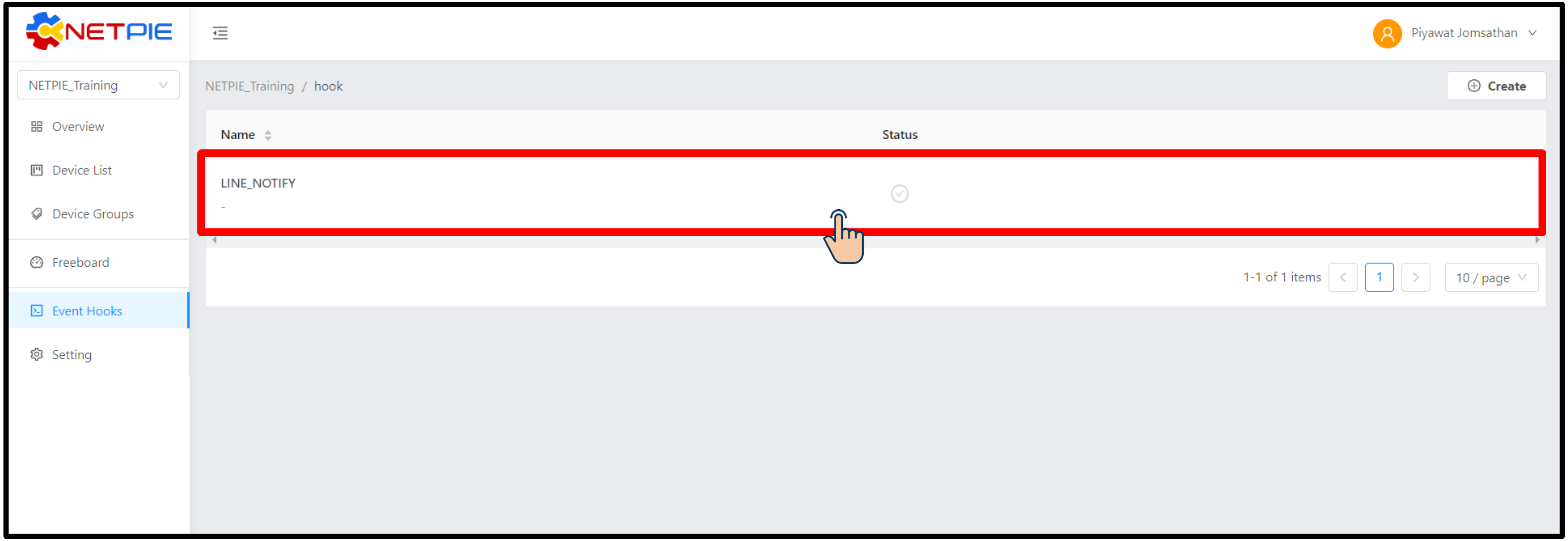


Create Event Hook



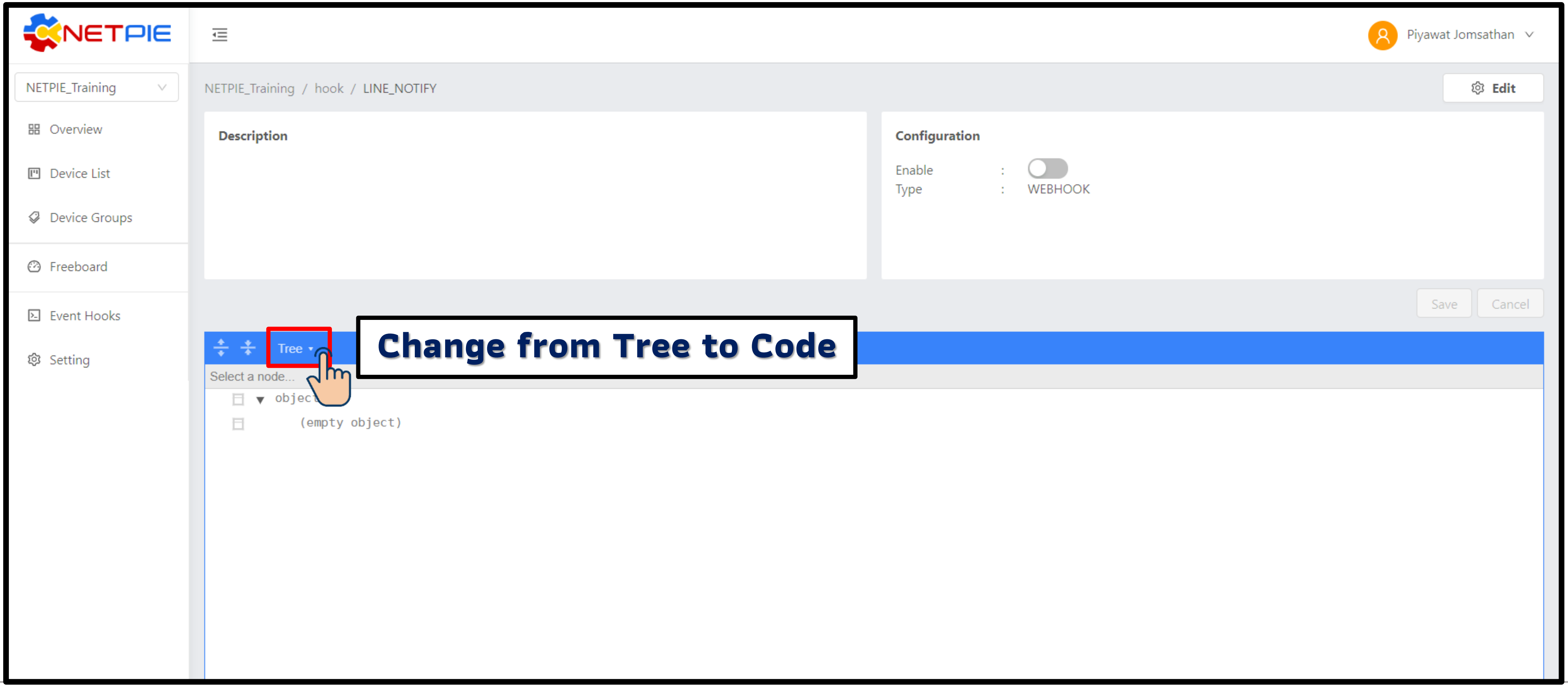
1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook

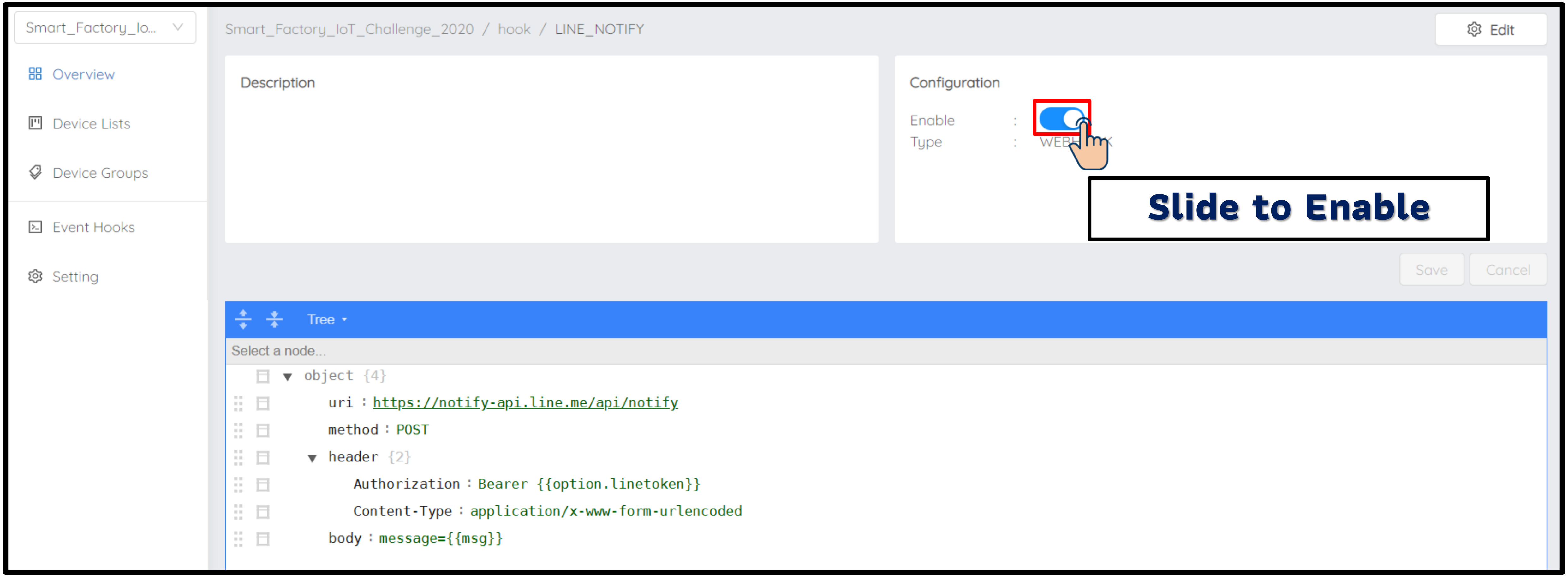
Event Hook Example of a Line Notify that specifies 4 Attributes

```
{
    "body": "message={{msg}}",
    "header": {
        "Authorization": "Bearer {{option.linetoken}}",
        "Content-Type": "application/x-www-form-urlencoded"
    },
    "method": "POST",
    "uri": "https://notify-api.line.me/api/notify"
}
```

Copy and paste to Event Hook on NETPIE2020

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Create Event Hook

Event Hook Example of a Line Notify that specifies 4 Attributes

```
{
  "body": "message={{msg}}",
  "header": {
     "Authorization": "Bearer {{option.linetoken}}",
     "Content-Type": "application/x-www-form-urlencoded"
  },
  "method": "POST",
  "uri": "https://notify-api.line.me/api/notify"
}
```

body : the message sent to destination

header : additional information such as Authorization, Content-Type

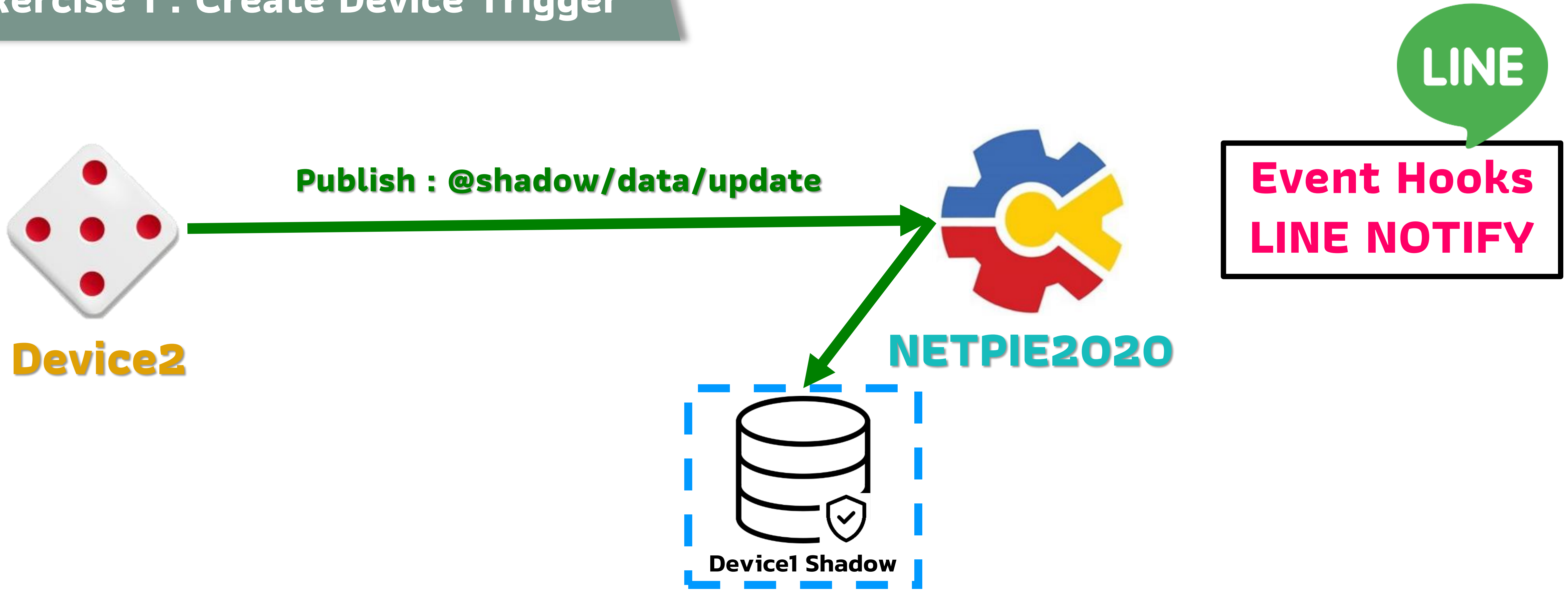
method : assign the method wanted by the receiving end GET, POST or PUT

uri : specify the Endpoint

*** In Event Hook, variables sent from Trigger can be referenced by using {{…}} symbols around them. For example, if we want to reference msg from Trigger, use {{msg}} Or in the option of linetoken, use {{option.linetoken}}

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger



Trigger : SHADOW UPDATE & DEVICE.STATUSCHANGE

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

Select Device1 to create Trigger

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

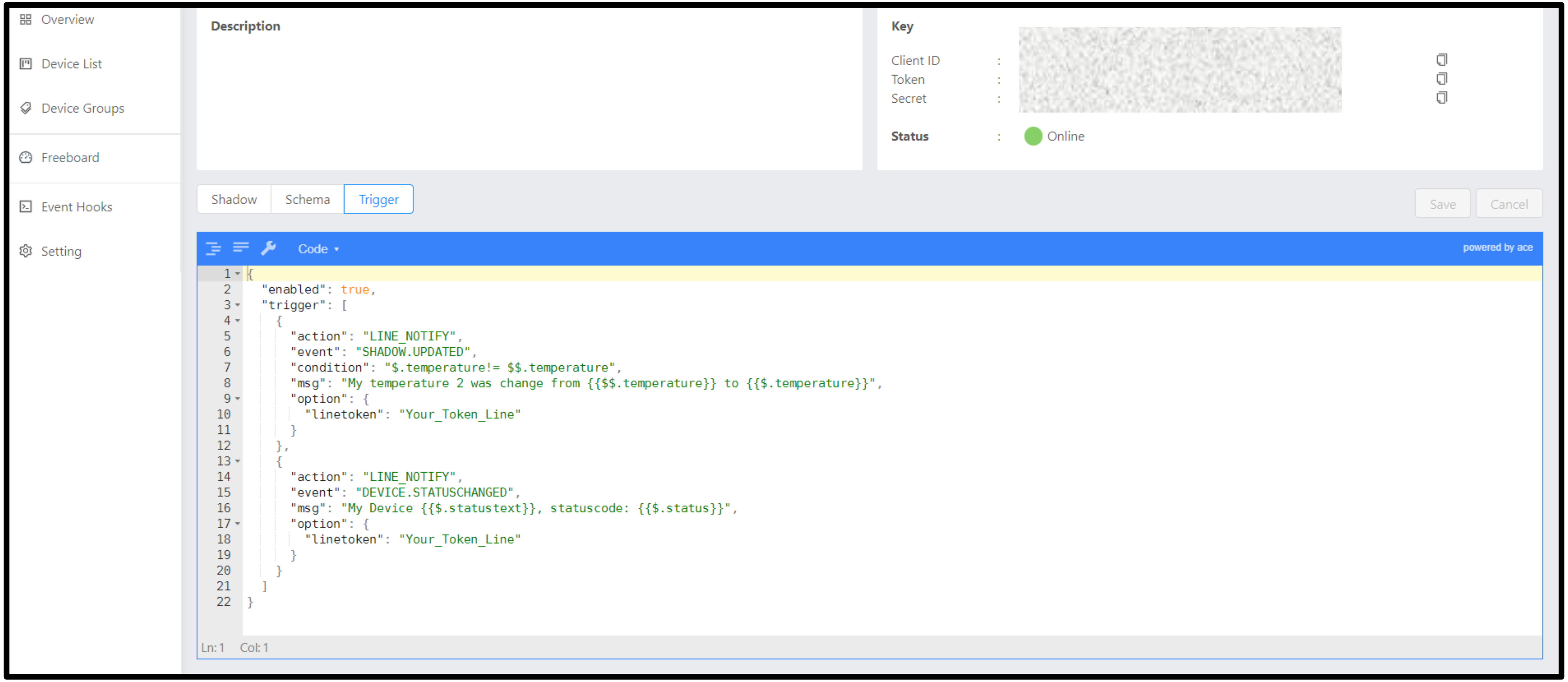
1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger


```
{
  "enabled": true,
  "trigger": [
    {
      "action": "LINE_NOTIFY",
      "event": "SHADOW.UPDATED",
      "condition": "$.temperature!= $$.temperature",
      "msg": "My temperature 2 was change from {{$$.temperature}} to {{$.temperature}}",
      "option": {
        "linetoken": "Your_Token_LINE"
      }
    },
    {
      "action": "LINE_NOTIFY",
      "event": "DEVICE.STATUSCHANGED",
      "msg": "My Device {{$.statustext}}, statuscode: {{$.status}}",
      "option": {
        "linetoken": "Your_Token_LINE"
      }
    }
  ]
}
```

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

```
{
 "enabled": true,
 "trigger": [
  {
   "action": "LINE_NOTIFY",
   "event": "SHADOW.UPDATED",
   "condition": "$.temperature!= $$.temperature",
   "msg": "My temperature 2 was change from {{$$.temperature}} to {{$.temperature}}",
   "option": {
    "linetoken": "Your_Token_LINE"
   }
  },
  {
   "action": "LINE_NOTIFY",
   "event": "DEVICE.STATUSCHANGED",
   "msg": "My Device {{$.statustext}}, statuscode: {{$.status}}",
   "option": {
    "linetoken": "Your_Token_LINE"
   }
  }
 ]
}
```

Turn on Trigger

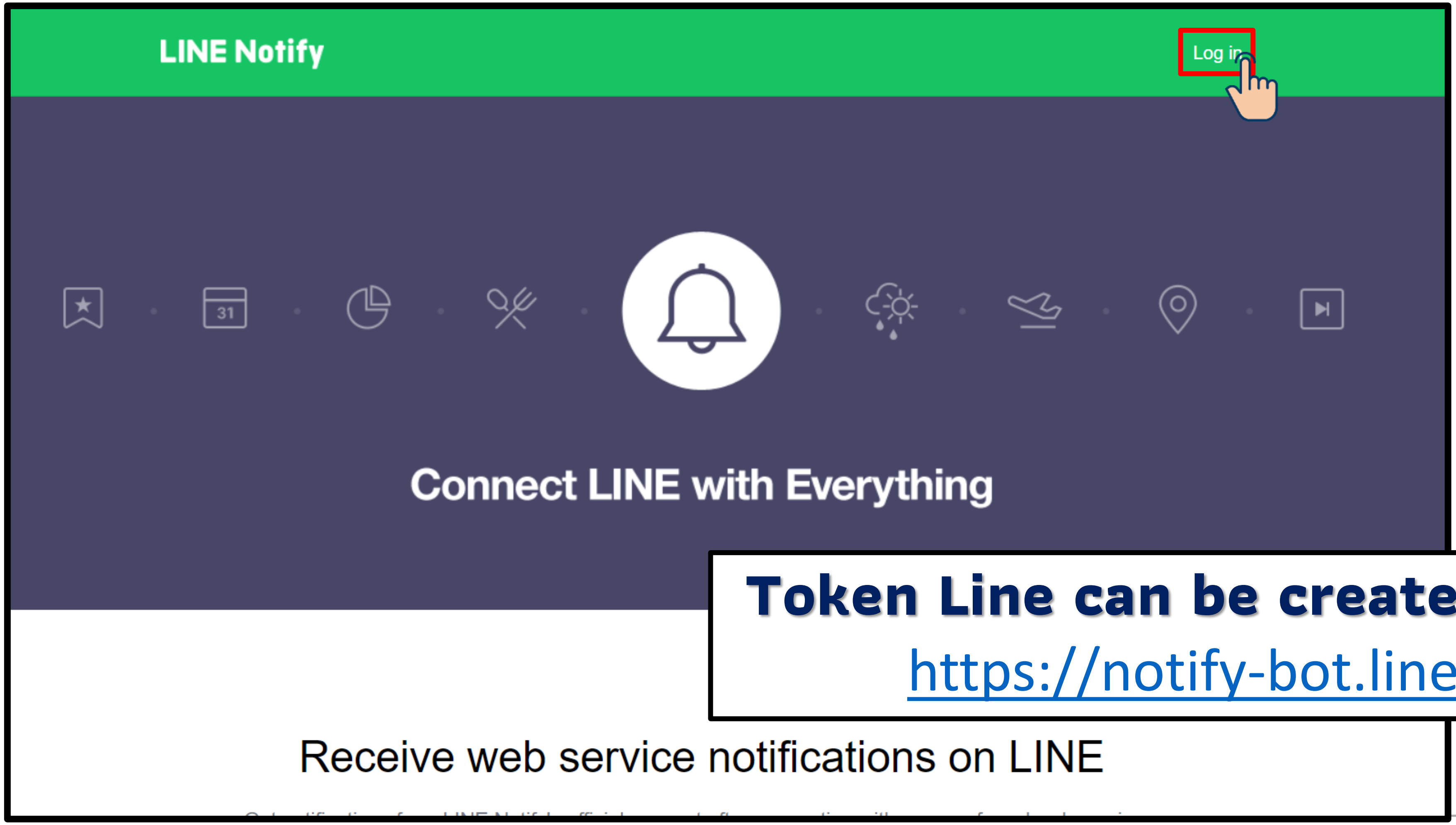
Copy this trigger code and paste to Device Trigger

Trigger Shadow Update : the condition is to take action whenever the temperature changes, with action : “LINE_NOTIFY” and the message is “My temperature 2 was changed from xx to xx.”

Trigger Status Change : The condition is to take action whenever device status changes, with action: “LINE_NOTIFY” and the message is “My device xx statuccode: xx”

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

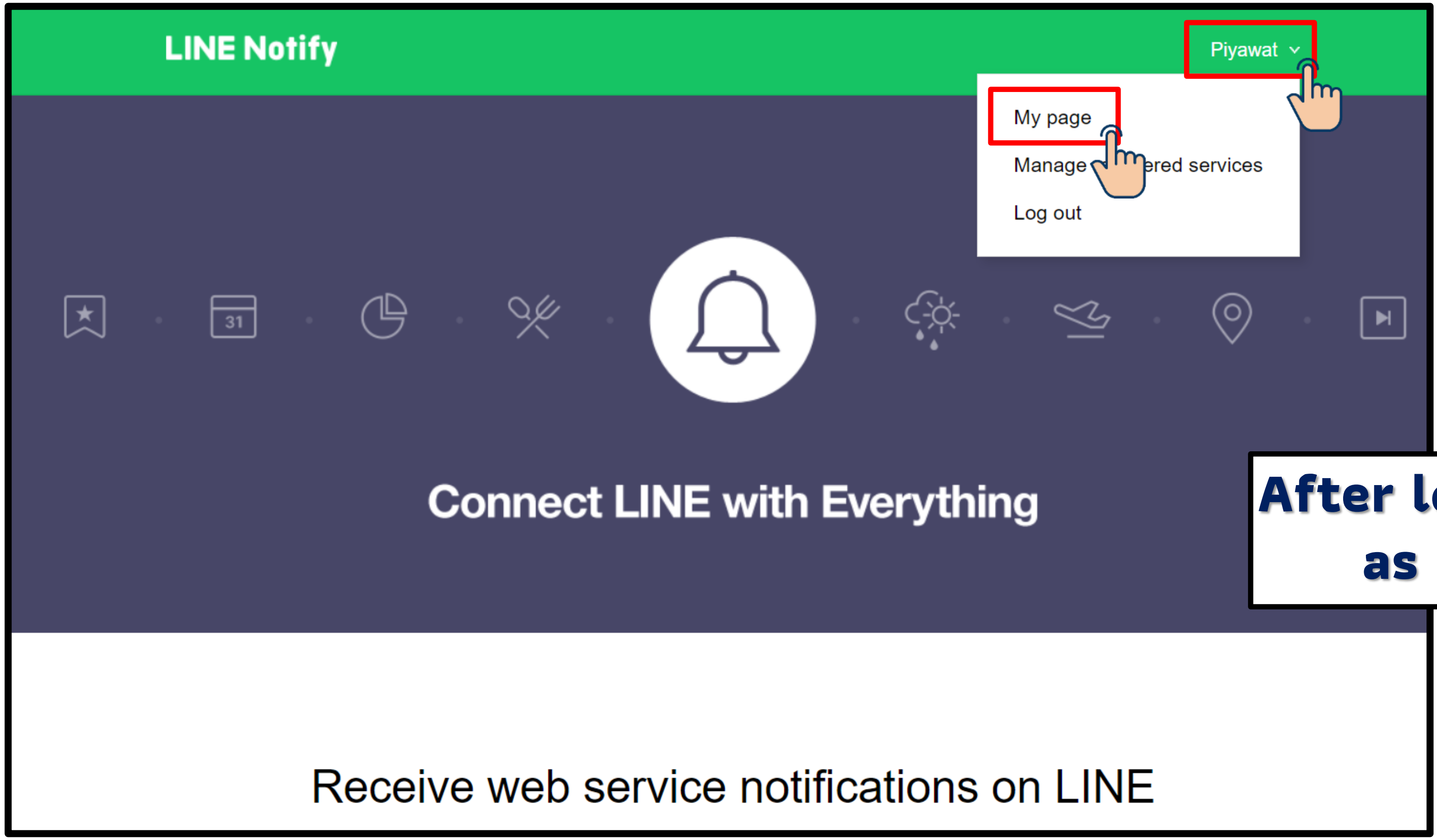


Token Line can be created at this link
https://notify-bot.line.me/en/

Exercise 1 : Create Device Trigger

Login to your LINE account using E-mail and Password

Exercise 1 : Create Device Trigger

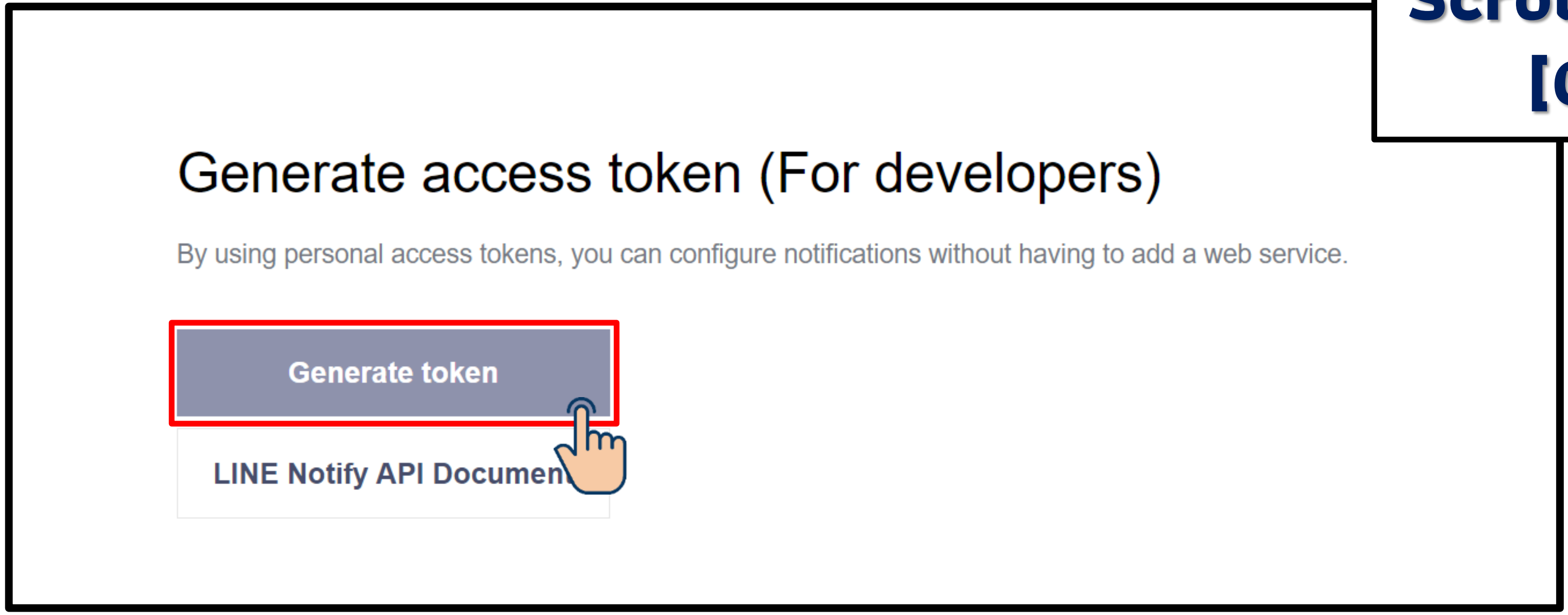


After login, select >> My page as shown in the figure

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

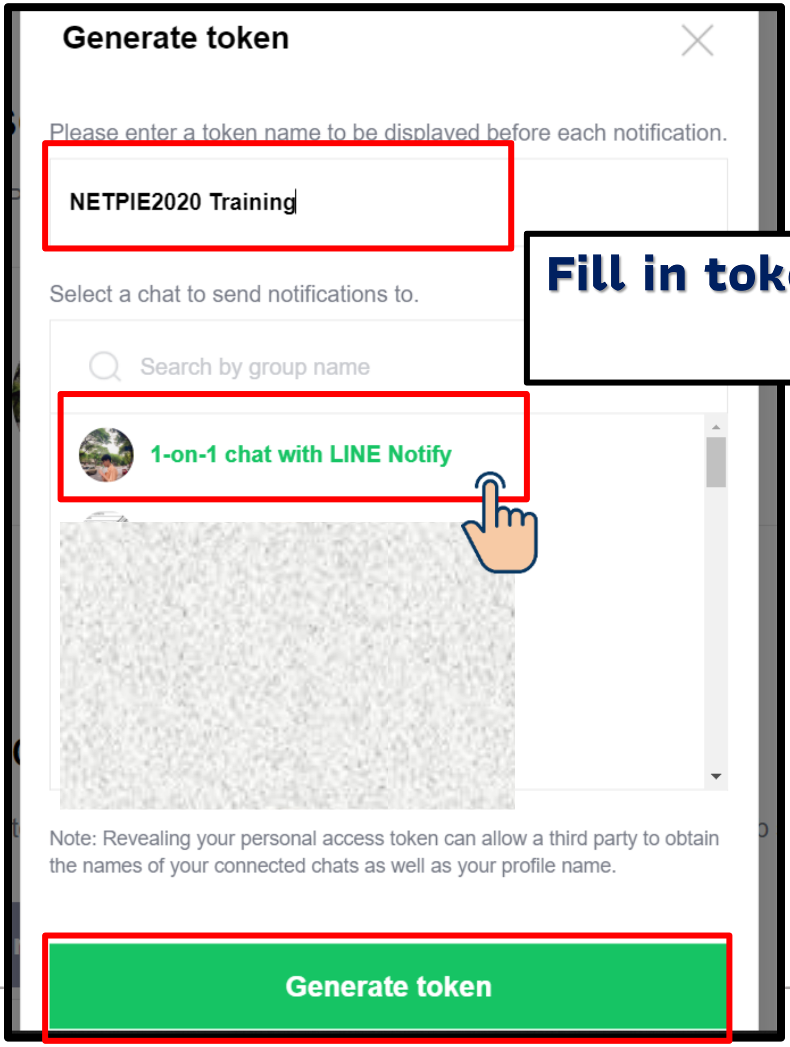
Exercise 1 : Create Device Trigger

Scroll down and click on [Generate Token]



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

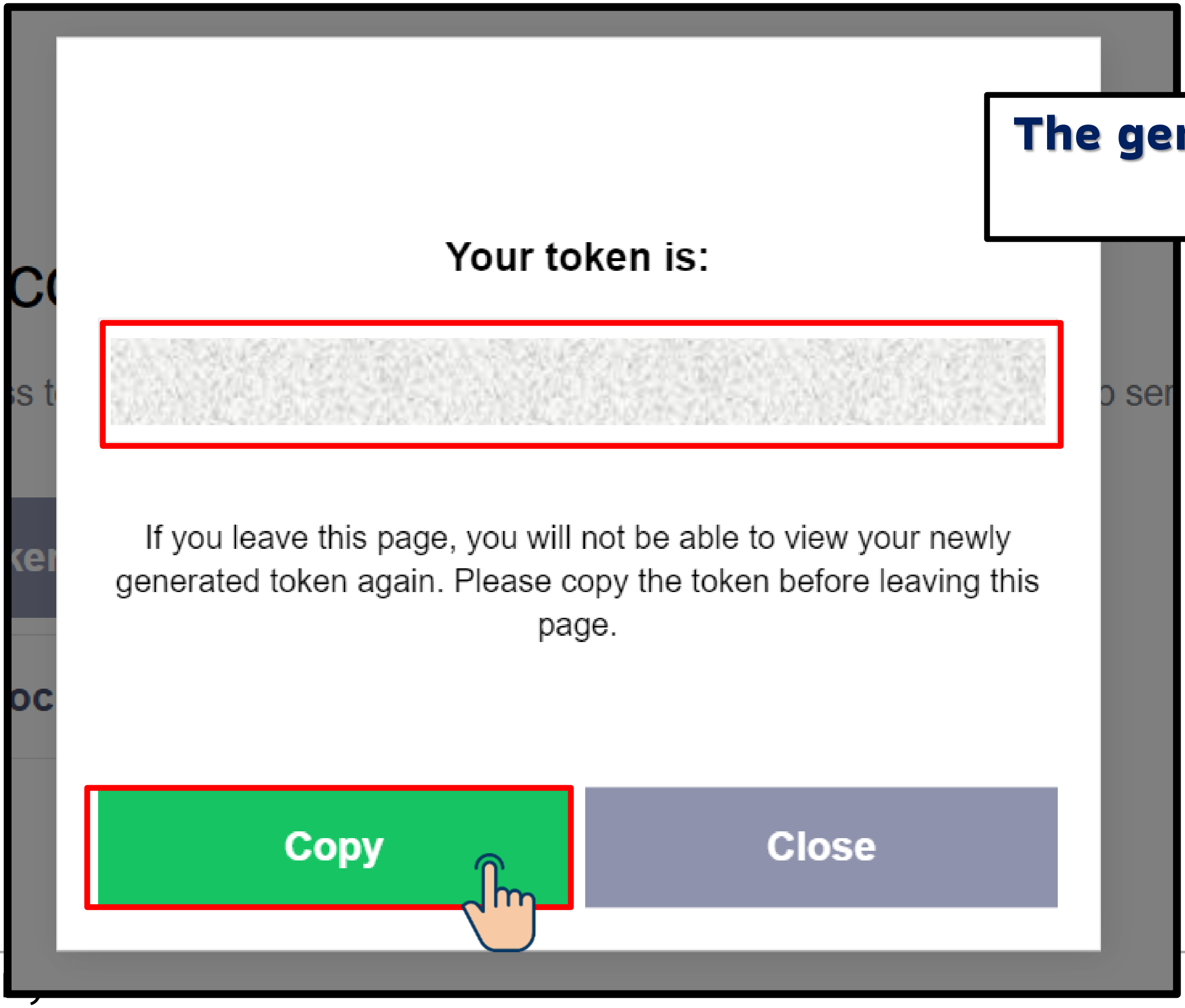
Exercise 1 : Create Device Trigger



Fill in token name, and select 1-on-1 chat with LINE notify.
Click [Generate Token]

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger



The generated token is displayed. Copy and paste to linetoken in Device Trigger

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

Shadow | Schema | Trigger

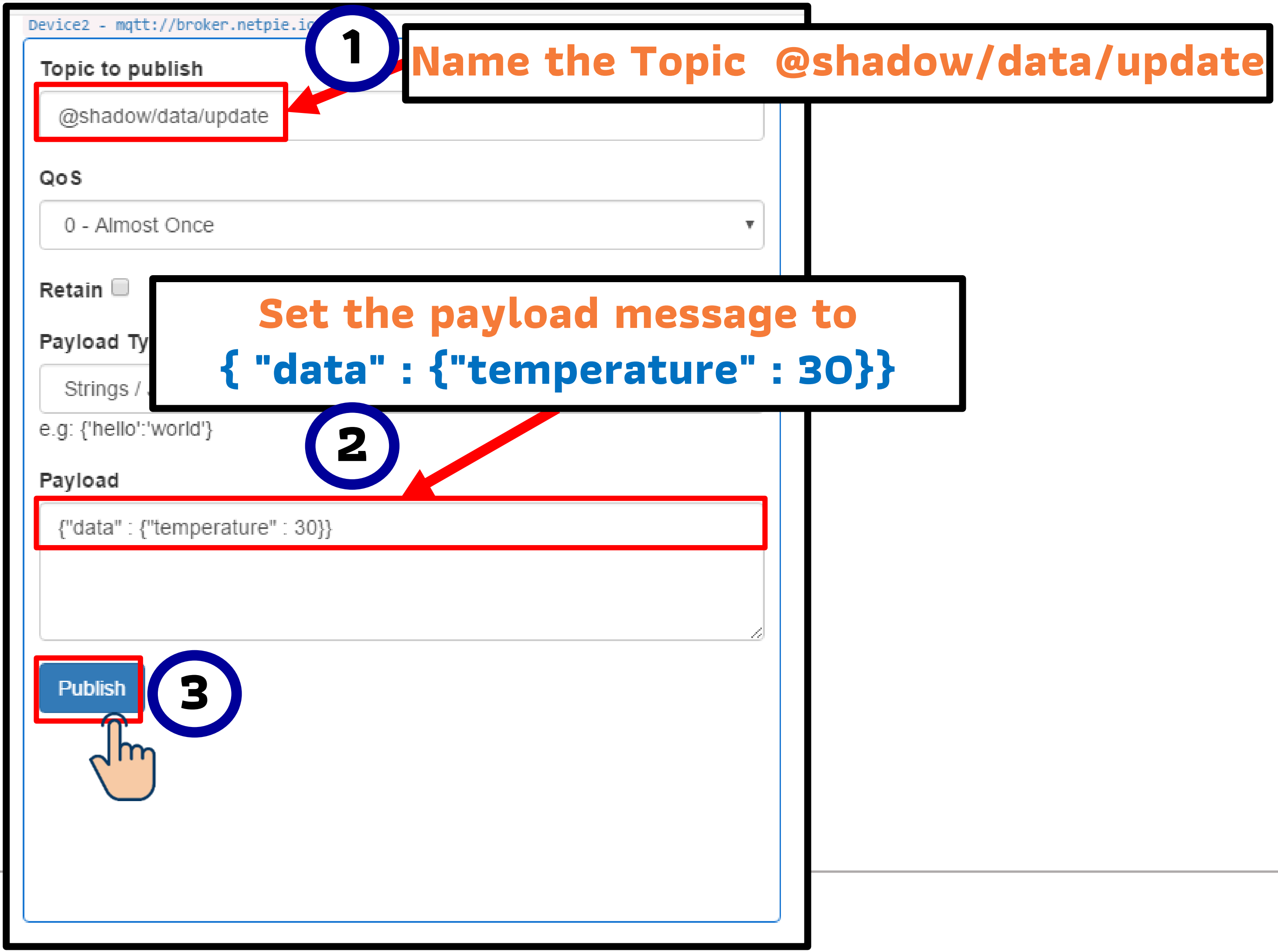
Save | Cancel

Code ▾ ... wered by ace

```
{
  "enabled": true,
  "trigger": [
    {
      "action": "LINE_NOTIFY",
      "event": "SHADOW.UPDATED",
      "condition": "$.temperature!= $$.temperature",
      "msg": "My temperature 2 was change from {{$$.temperature}} to {{$.temperature}}",
      "option": {
        "linetoken": "Your_Token_LINE"
      }
    },
    {
      "action": "LINE_NOTIFY",
      "event": "DEVICE.STATUSCHANGED",
      "msg": "My Device {{$.statustext}}, statuscode: {{$.status}}",
      "option": {
        "linetoken": "Your_Token_LINE"
      }
    }
  ]
}
```

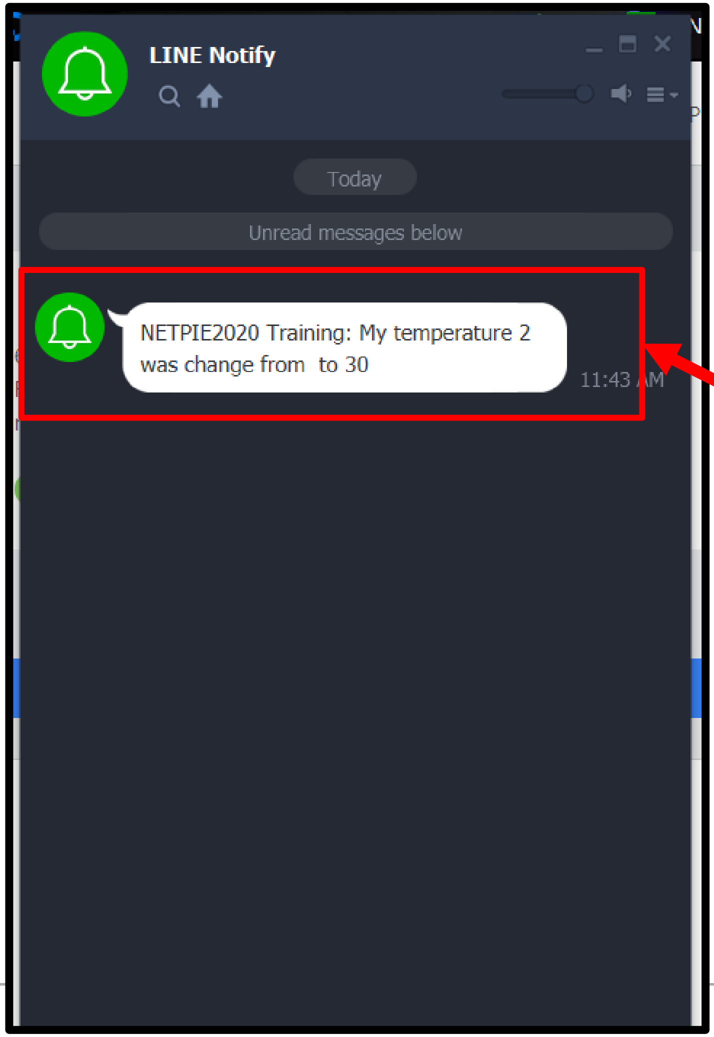
Ln: 22 Col: 2

Exercise 1 : Create Device Trigger



1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

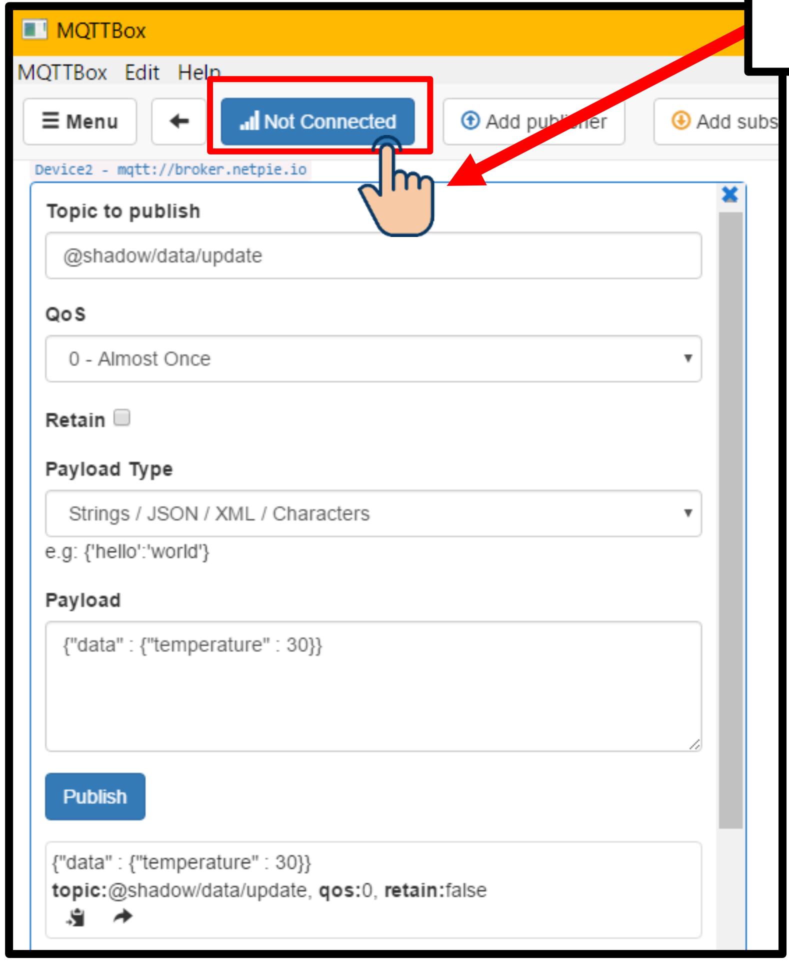


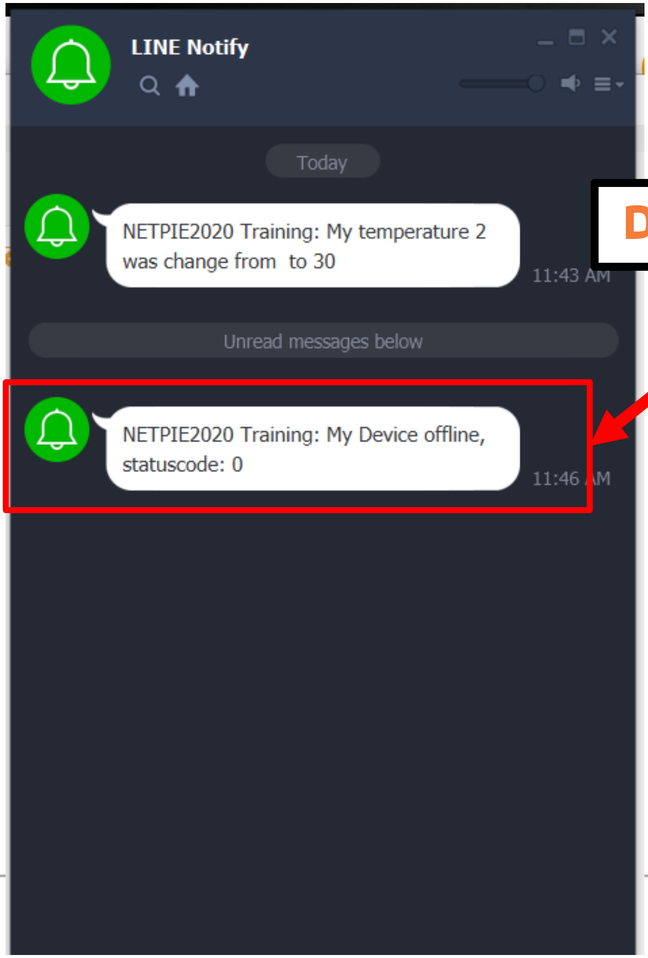
Temperature change notification from LINE

1 - Data Management in NETPIE2020 (Trigger & Event Hook)

Exercise 1 : Create Device Trigger

Device status change test





Device status change notification LINE

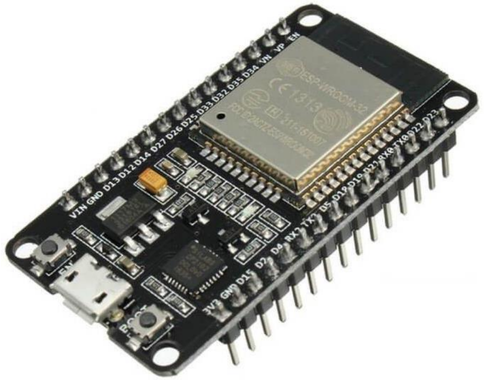
2

MQTT Library in Arduino IDE (Pubsubclient)

where things chat

2 - MQTT Library in Arduino IDE (Pubsubclient)

In order for ESP32 or ESP8266 to connect to NETPIE2020, it needs to be programmed on the Arduino IDE to connect ESP32 or ESP8266 to internet and then to NETPIE2020.



Arduino IDE

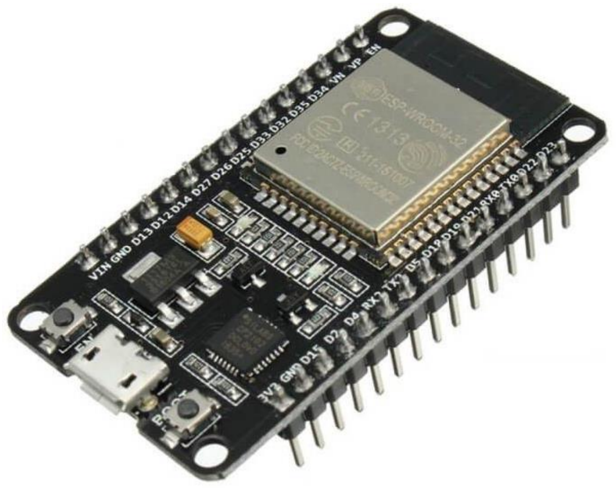
ESP32 or ESP8266

Internet connection

through WiFi

NETPIE2020

2 - MQTT Library in Arduino IDE (Pubsubclient)



ESP32 or ESP8266

In order for ESP32 or ESP8266 to connect to NETPIE2020, the following libraries are needed

1. WiFi

Used to allow ESP32 or ESP8266 to connect to the Internet via WiFi network.

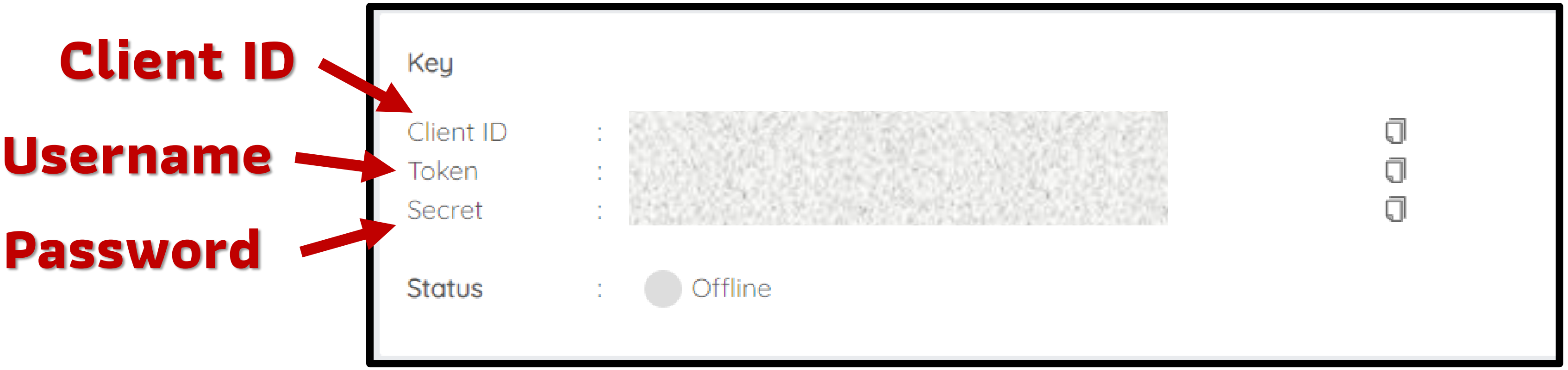
2. PubSubClient

Used for ESP32 or ESP8266 to connect and communicate on NETPIE 2020 platform. [MQTT Protocol]

2 - MQTT Library in Arduino IDE (Pubsubclient)

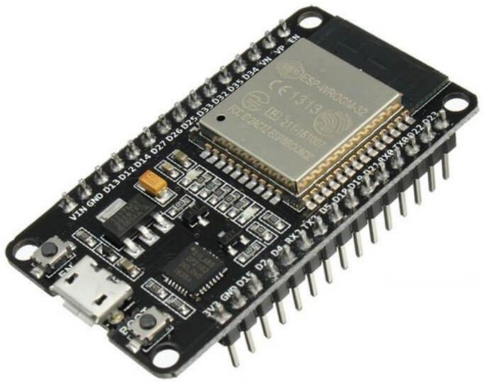
NETPIE2020 connection requires 4 parameters as follows:

1. **Host : broker.netpie.io**
2. **Client ID : Client ID of Device created in NETPIE2020 portal.**
3. **Username : Token of Device created in NETPIE2020 portal.**
4. **Password : Secret of Device created n NETPIE2020 portal.**
   [used for added security]



2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266



ESP32 or ESP8266

Connect

NETPIE2020

Please check condition
1. Install Arduino IDE
2. Install ESP32 or ESP8266 Hardware.
3. Install PubSubClient Library

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266

Coding in connect to NETPIE2020 consists of 3 sections

1 **Section 1 : Library and variable declaration**

Call to various libraries.

```
#include <WiFi.h>
#include <PubSubClient.h>
```

Variable declaration for connecting to WiFi

```
const char* ssid = "Your WiFi SSID";
const char* password = "Your Password";
```

Variable declaration for NETPIE2020 connection

```
const char* mqtt_server = "broker.netpie.io";
const int mqtt_port = 1883;
const char* mqtt_Client = "Your Client-ID";
const char* mqtt_username = "Your Token";
const char* mqtt_password = "Your Secret";
```

Create instances for NETPIE2020 connection

```
WiFiClient espClient;
PubSubClient client(espClient);
```

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266

2

Section 2 : functions

Function for NETPIE2020 connection

```
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting NETPIE2020 connection...");
    if (client.connect(mqtt_Client, mqtt_username, mqtt_password)) {
      Serial.println("NETPIE2020 connected");
    }
    else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println("try again in 5 seconds");
      delay(5000);
    }
  }
}
```

Enter NETPIE2020 connection.

- If connected successfully, it will display "connected".
- If the connection is not successful, it will display “failed…” and automatically reconnect.

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266

2 **Section 2 : functions**

Performing initialization

In this function, it will connect to WiFi and NETPIE2020 according to the settings.

```
void setup() {
  Serial.begin(115200);
  Serial.println("Starting...");
  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(1000);
      Serial.print(".");
    }
  }
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  client.setServer(mqtt_server, mqtt_port);
}
```

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266

3 **Section 3 : loop() function**

Main function that runs iteratively

```
void loop() {

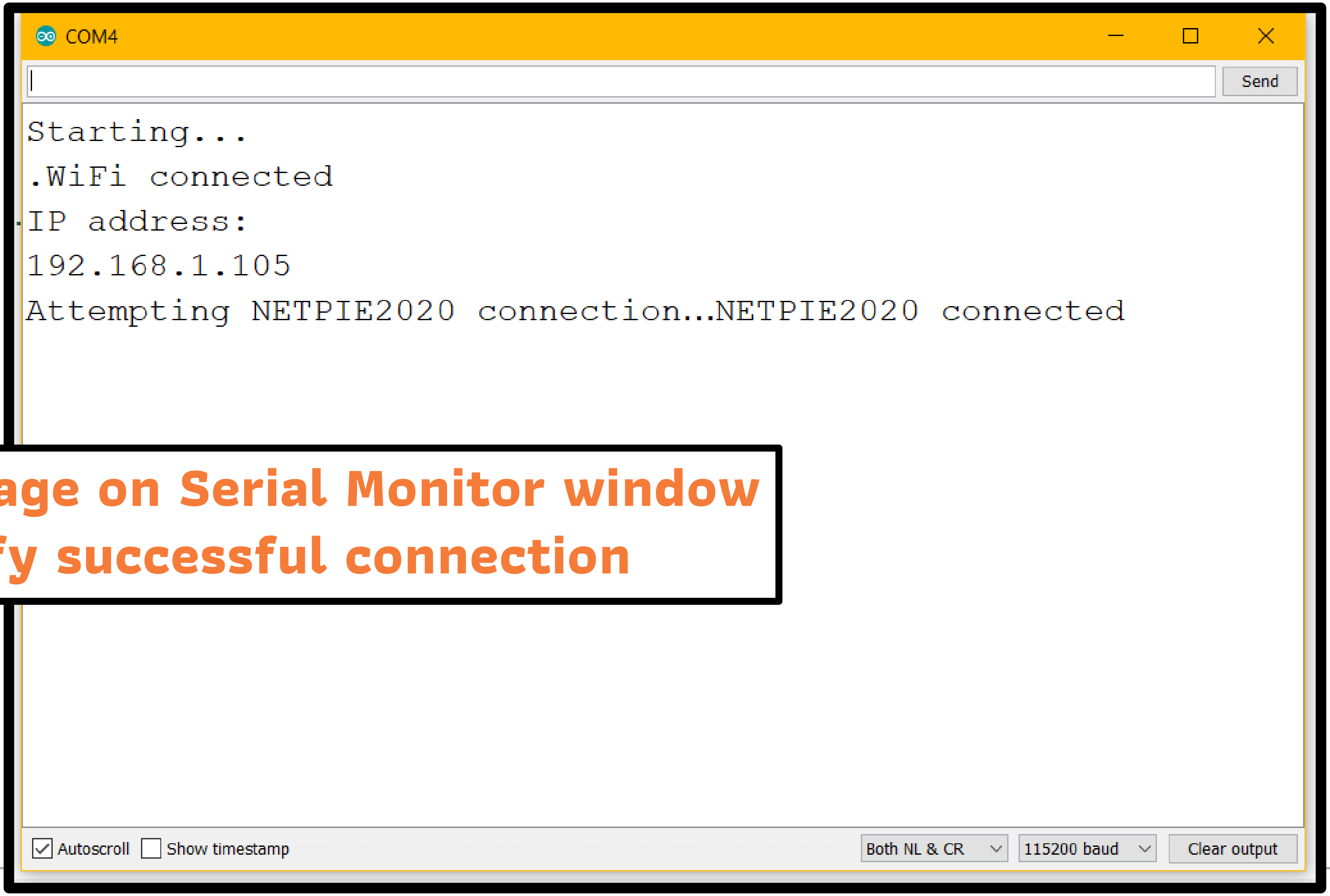
 if (!client.connected()) {
   reconnect();
 }
 client.loop();

}
```

Command set that maintains the connection status and functions of NETPIE2020.

2 - MQTT Library in Arduino IDE (Pubsubclient)

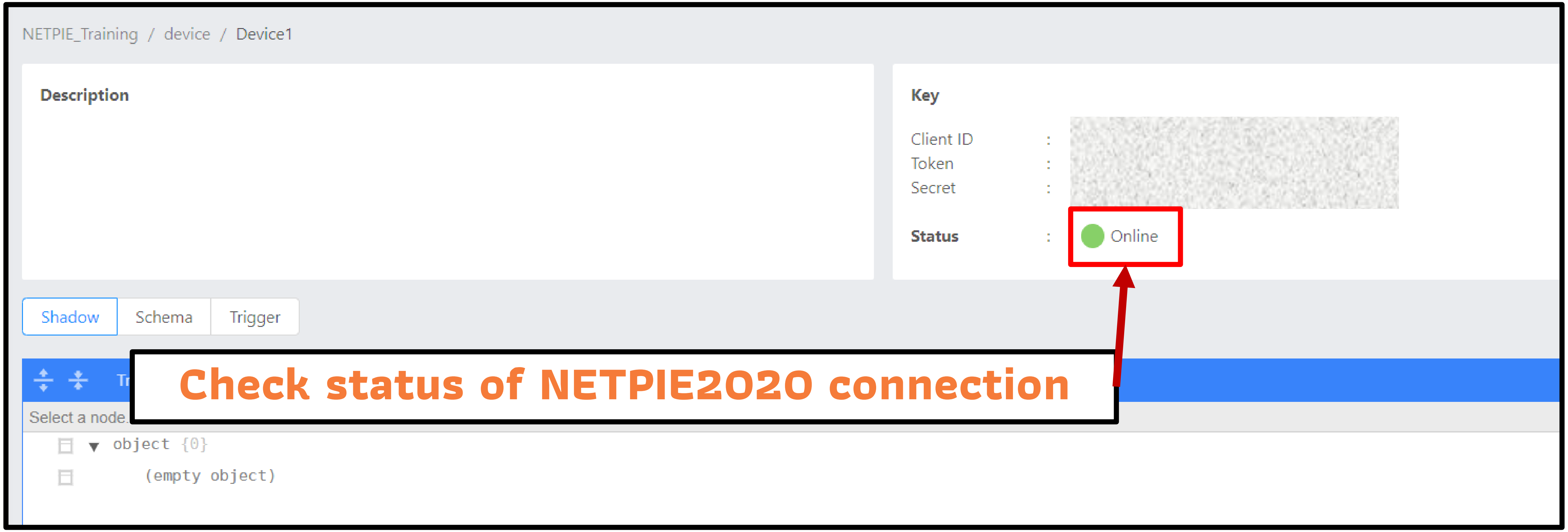
Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266



Check message on Serial Monitor window to verify successful connection

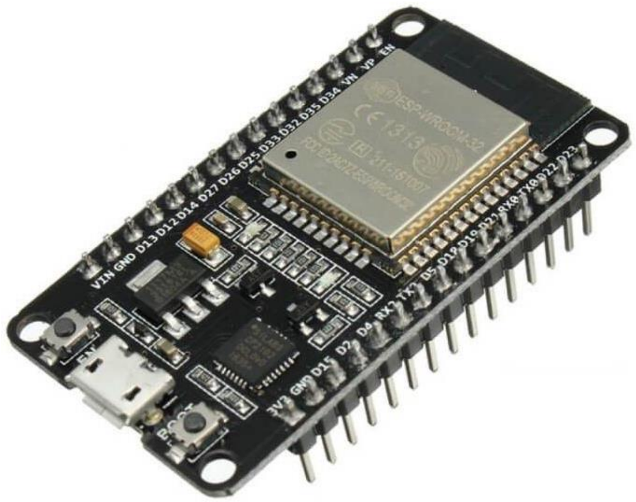
2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 2 : Connecting the NETPIE2020 with ESP32 or ESP8266



2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 3 : Communication on NETPIE2020 with ESP32 or ESP8266 (Publish)



ESP32 or ESP8266

"Hello NETPIE2020"

NETPIE2020

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 3 : Communication on NETPIE2020 with ESP32 or ESP8266 (Publish)

Coding in Exercise2 has the beginning part WiFi and NETPIE2020 connection similar Exercise2. Only the void loop() part is added.

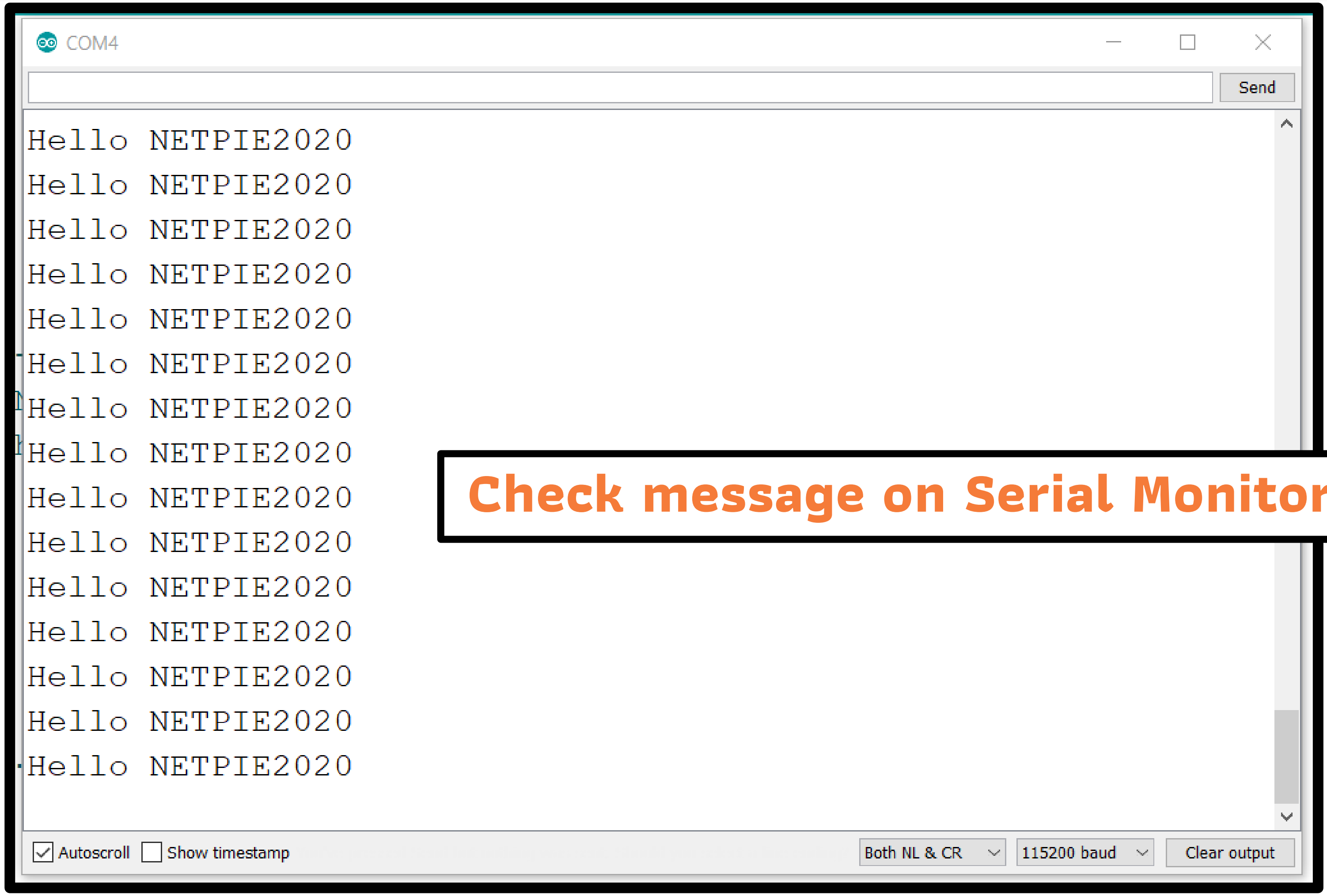
client.publish is a command to publish messages to NETPIE2020 using format
client.publish ("Topic", "Message");

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

  client.publish("@msg/test", "Hello NETPIE2020");
  Serial.println("Hello NETPIE2020");
  delay(2000);
}
```

2 - MQTT Library in Arduino IDE (Pubsubclient)

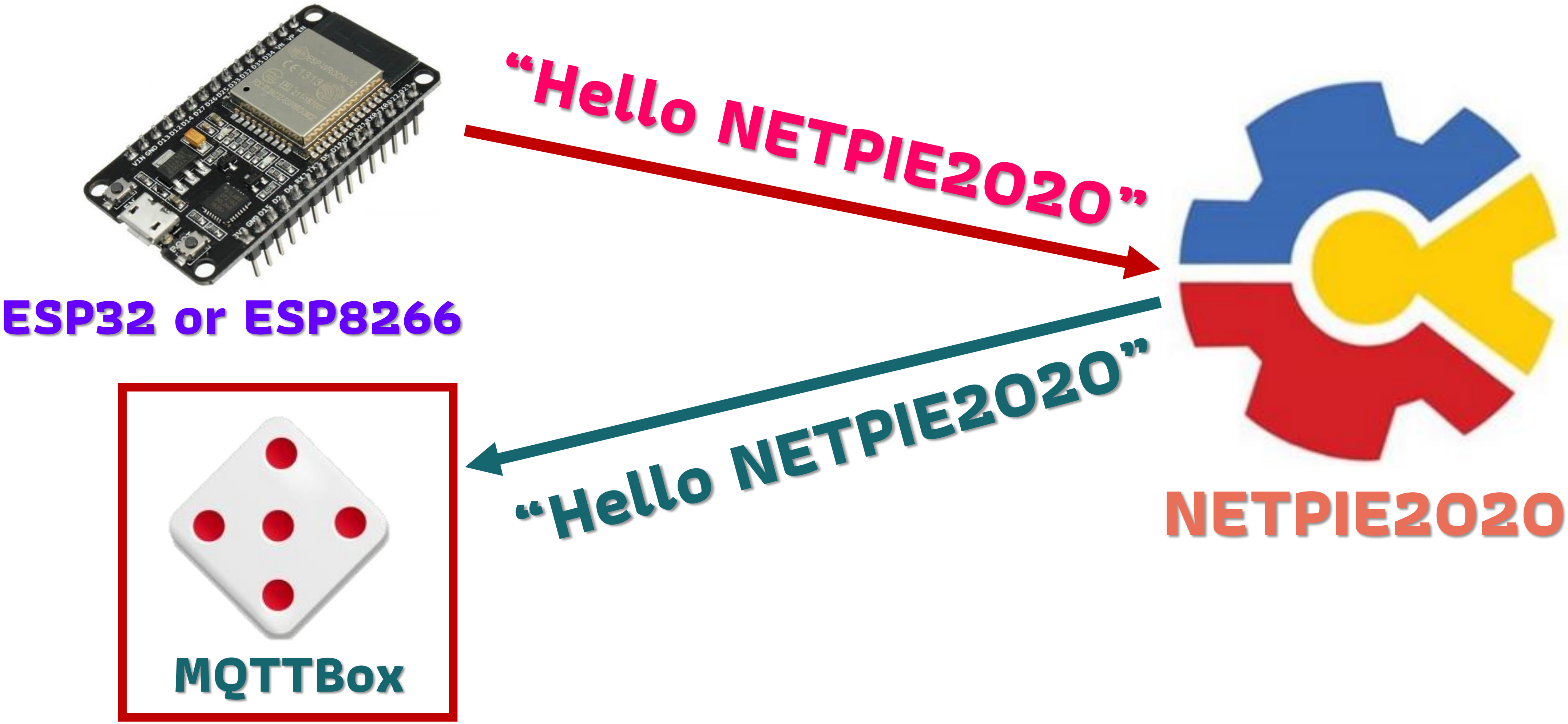
Exercise 3 : Communication on NETPIE2020 with ESP32 or ESP8266 [Publish]



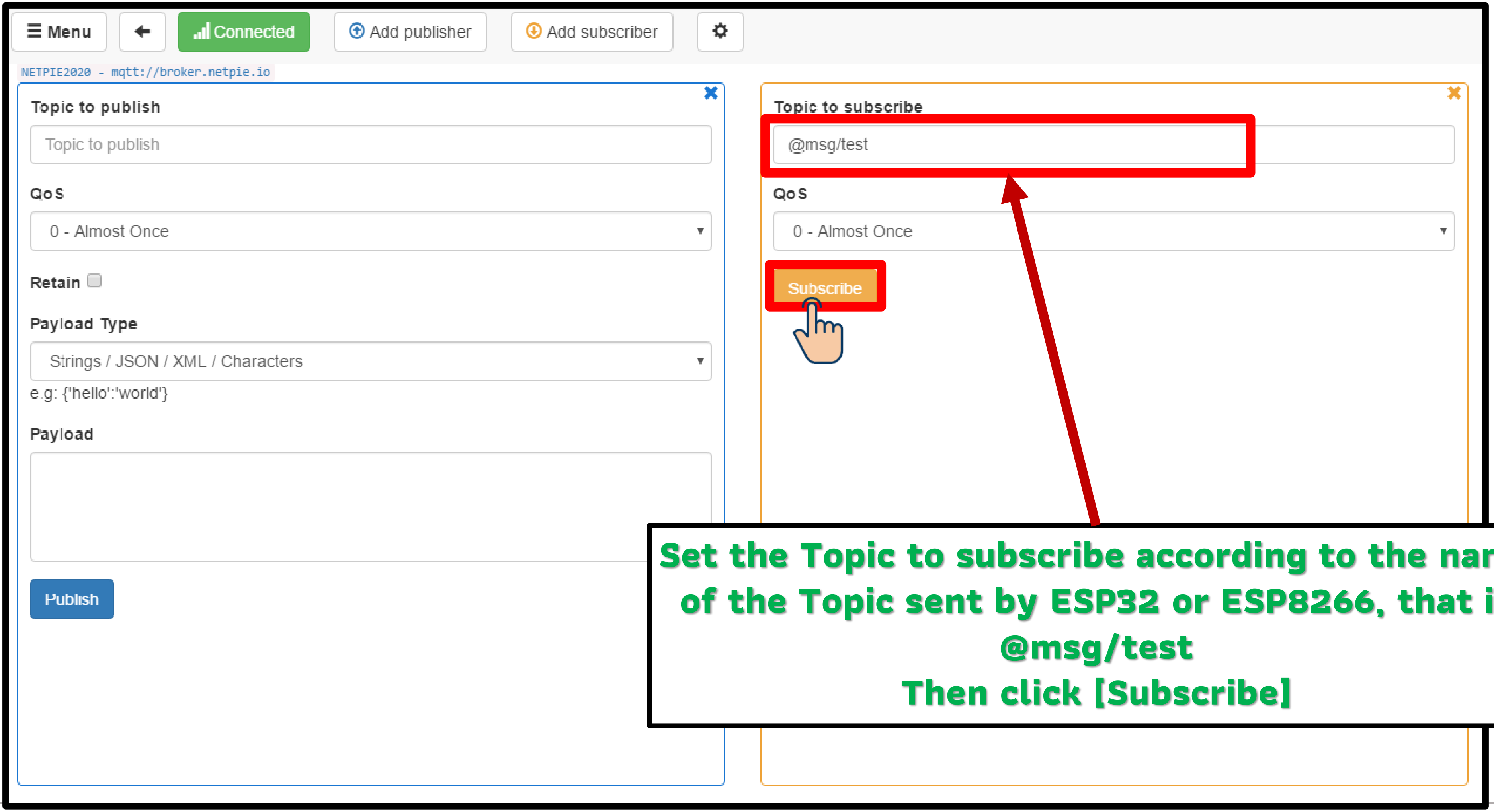
Check message on Serial Monitor

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 4 : Communication on NETPIE2020 with ESP32 or ESP8266 [Subscribe]



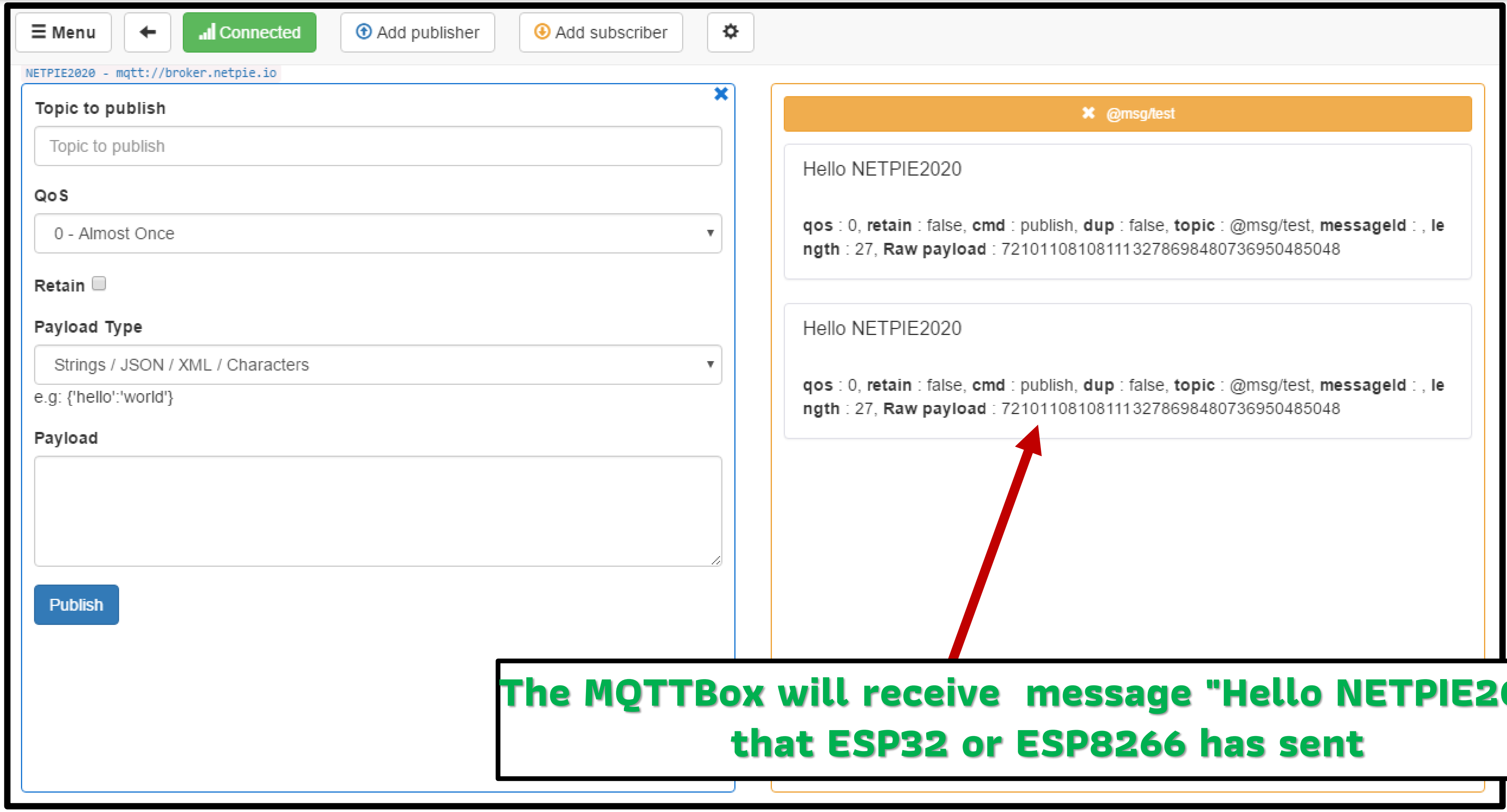
Exercise 4 : Communication on NETPIE2020 with ESP32 or ESP8266 [Subscribe]



Set the Topic to subscribe according to the name of the Topic sent by ESP32 or ESP8266, that is @msg/test
Then click [Subscribe]

2 - MQTT Library in Arduino IDE (Pubsubclient)

Exercise 4 : Communication on NETPIE2020 with ESP32 or ESP8266 [Subscribe]



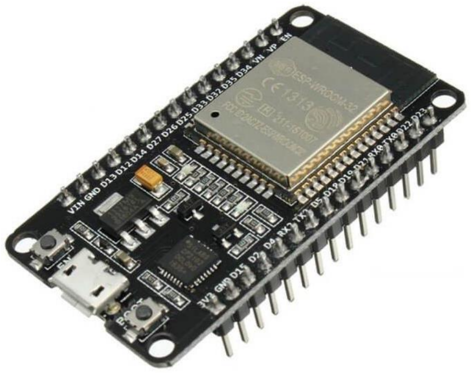
The MQTTBox will receive message "Hello NETPIE2020" that ESP32 or ESP8266 has sent

3

Freeboard in NETPIE2020 (Control Widget)

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



@shadow/data/update

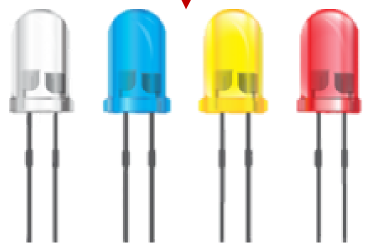
Publish : { "data": { "temperature" : 24, "light" : 120 }}

LED : ON/OFF

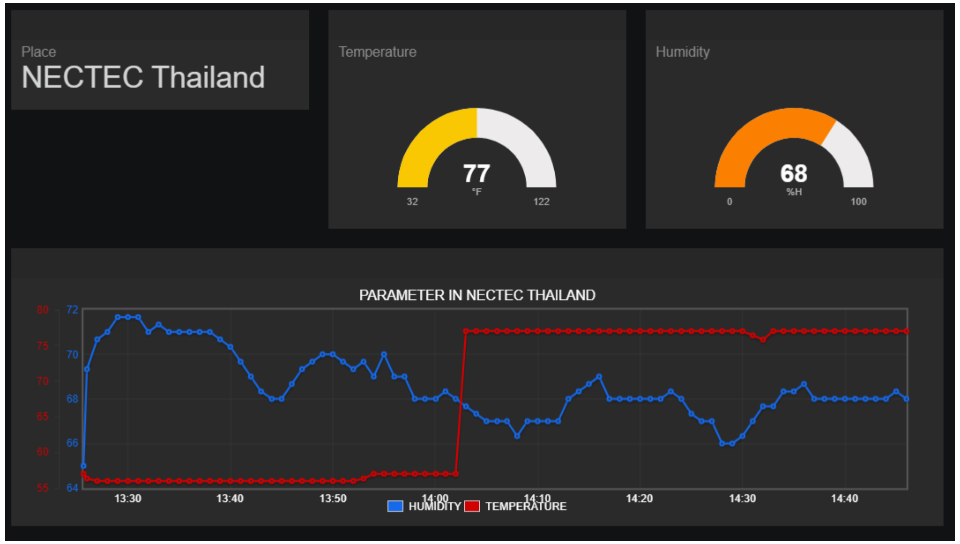
NETPIE2020

ESP32 or ESP8266

On ,Off



Freeboard

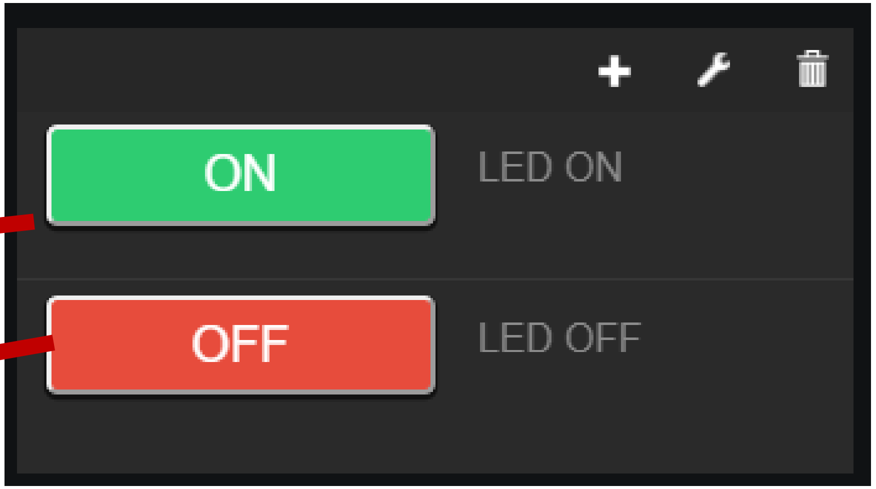


3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

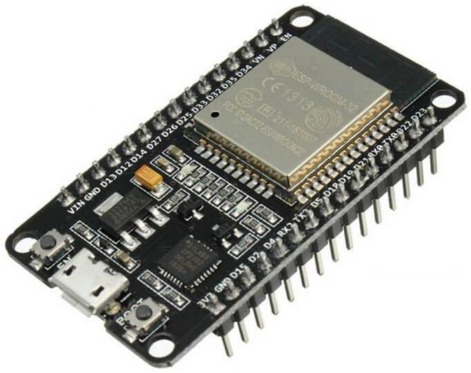
LED on ESP32 or ESP8266 can be controlled via Freeboard by 2 methods

1. Command via Shadow



@shadow/data/update
“led” : “on”

@shadow/data/update
“led” : “off”



ESP32 or ESP8266

“led” : on

Device Shadow

3 - Freeboard in NETPIE2020 (Control Widget)

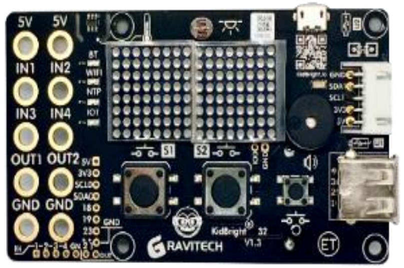
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

LED on ESP32 or ESP8266 can be controlled via Freeboard by 2 methods

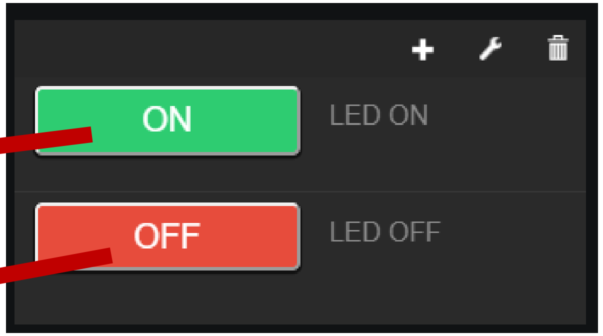
1. Command via Shadow

Message sent via shadow has the following format

b'{"deviceid":"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx","data":{"led":"on"},"rev":3,"modified":1579864929867}'

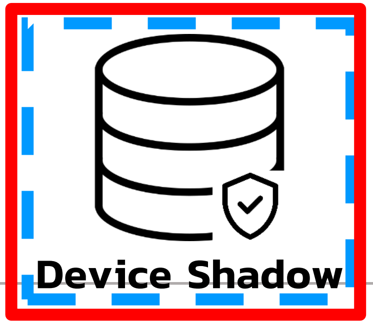


@shadow/data/update
"led" : "on"



@shadow/data/update
"led" : "off"

On ,Off

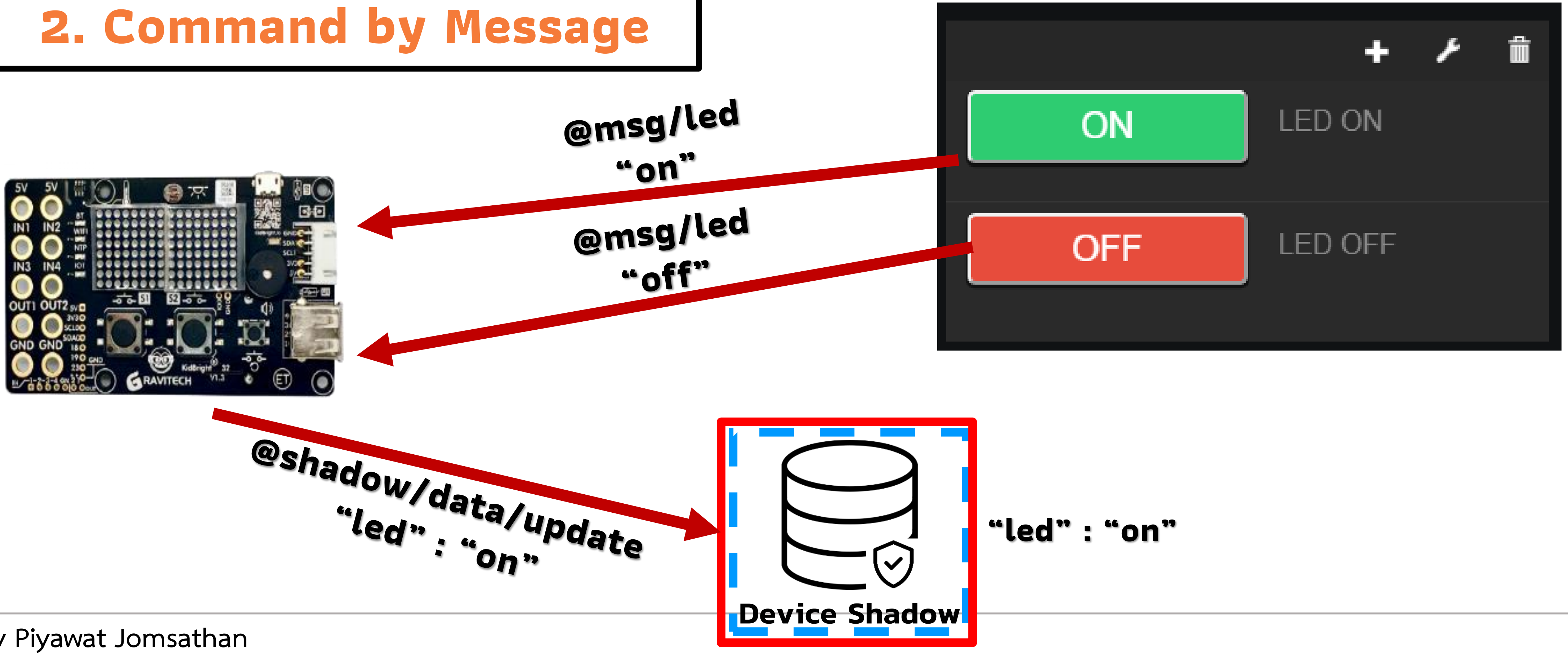


3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

LED on ESP32 or ESP8266 can be controlled via Freeboard by 2 methods

2. Command by Message



3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

2 **Part 2 various functions**

In Coding

MQTT connection function

```
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection…");
    if (client.connect(mqtt_Client, mqtt_username, mqtt_password)) {
      Serial.println("connected");
      client.subscribe("@msg/led");
    }
    else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println("try again in 5 seconds");
      delay(5000);
    }
  }
}
```

Subscribe Topic sent by Freeboard

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

2 **Part 2 various functions**

MQTT message receive function

Commands to display MQTT topic and message

LED on / off condition :starting by checking whether topic is @ msg / led or not And check the received message whether it is on or off .Because the LED on the ESP8266 or ESP32 is a pull-up circuit. Thus we must send 0 to the LED pin to turn on, and send 1 to turn off.

```
void callback(char* topic, byte* payload, unsigned int length) {
 Serial.print("Message arrived [");
 Serial.print(topic);
 Serial.print("] ");
 String message;
 for (int i = 0; i < length; i++) {
   message = message + (char)payload[i];
 }
 Serial.println(message);

 if(String(topic) == "@msg/led") {
   if (message == "on"){
     digitalWrite(LED1,0);
     client.publish("@shadow/data/update", "{\"data\" : {\"led\" : \"on\"}}");
     Serial.println("LED ON");   }
   else if (message == "off") {
     digitalWrite(LED1,1);
     client.publish("@shadow/data/update", "{\"data\" : {\"led\" : \"off\"}}");
     Serial.println("LED OFF");  }
} }
```

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

2 Part 2 various functions

```
void setup() {
  Serial.begin(115200);
  Wire1.begin(4, 5);
  pinMode(LED,OUTPUT);
  digitalWrite(LED,1);
  Serial.println("Starting...");
  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(1000);
      Serial.print(".");
    }
  }
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  client.setServer(mqtt_server, mqtt_port);
  client.setCallback(callback)
}
```

Setup function

Declare LED pin as OUTPUT
And start by turning it off

Assign callback function to MQTT

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

3 **Part 3 loop function**

```
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  temperature = readTemperature();
  light = readlight();
  String place = "NECTEC";


  long now = millis();
  if (now - lastMsg > 5000) {
    lastMsg = now;
    ++value;
    String data = "{\"data\": {\"light\":" + String(light) + ", \"temperature\":" + String(temperature) + ", \"place\":
\"" +  String(place) + "\"}}";
    Serial.println(data);
    data.toCharArray(msg, (data.length() + 1));
    client.publish("@shadow/data/update", msg);  }
  delay(1);
}
```

Use millis() for timing instead of delay() to avoid blocking the execution while waiting Freeboard message

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

Device Schema in JSON format

Add code on Schema for LED status to record

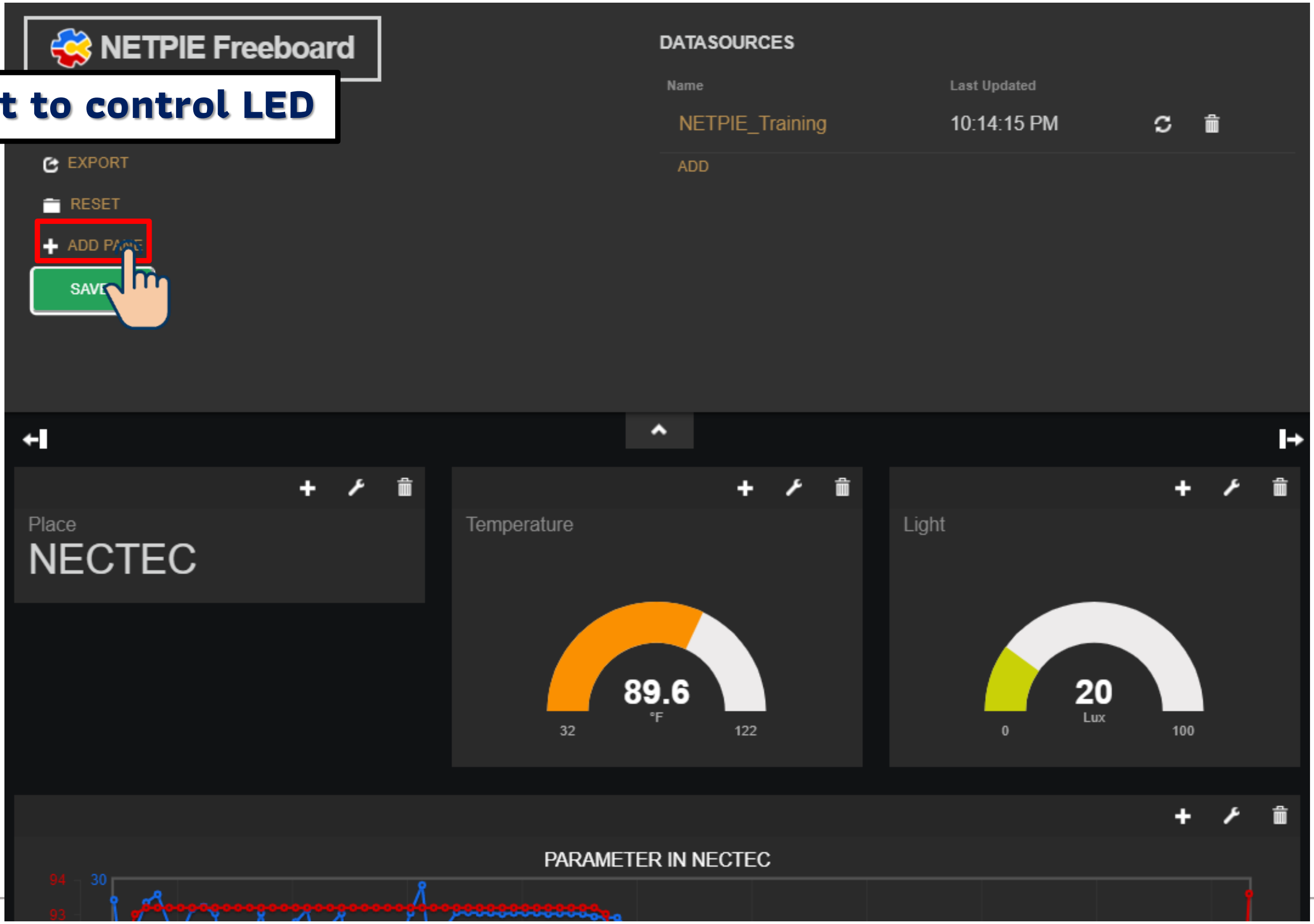
```
{
 "additionalProperties": false,
 "properties": {
  "light": {
   "operation": {
    "store": {
     "ttl": "7d"
    }
   },
   "type": "number"
  },
  "temperature": {
   "operation": {
    "store": {
     "ttl": "7d"
    },
    "transform": {
     "expression": "[($.temperature)*1.8] + 32"
    }
   },
   "type": "number"
  },
  "place": {
   "operation": {
    "store": {
     "ttl": "7d"
    }
   },
   "type": "string"
  },
  "led": {
   "operation": {
    "store": {
     "ttl": "7d"
    }
   },
   "type": "string"
  }
 }
}
```

Record **led** data with property
- Store for 7 days
- Variable type is string

3 - Freeboard in NETPIE2020 (Control Widget)

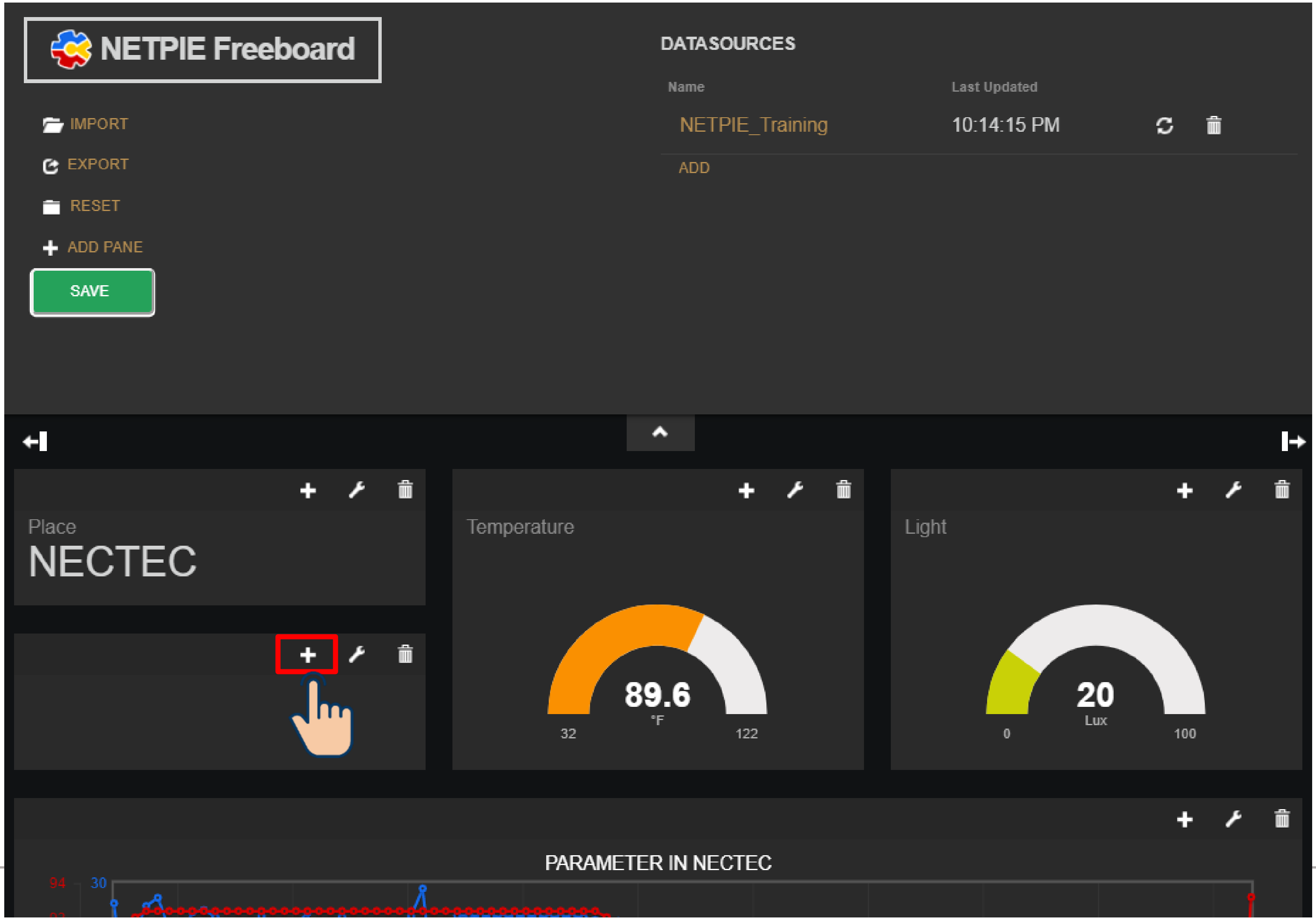
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard

Create a button widget to control LED



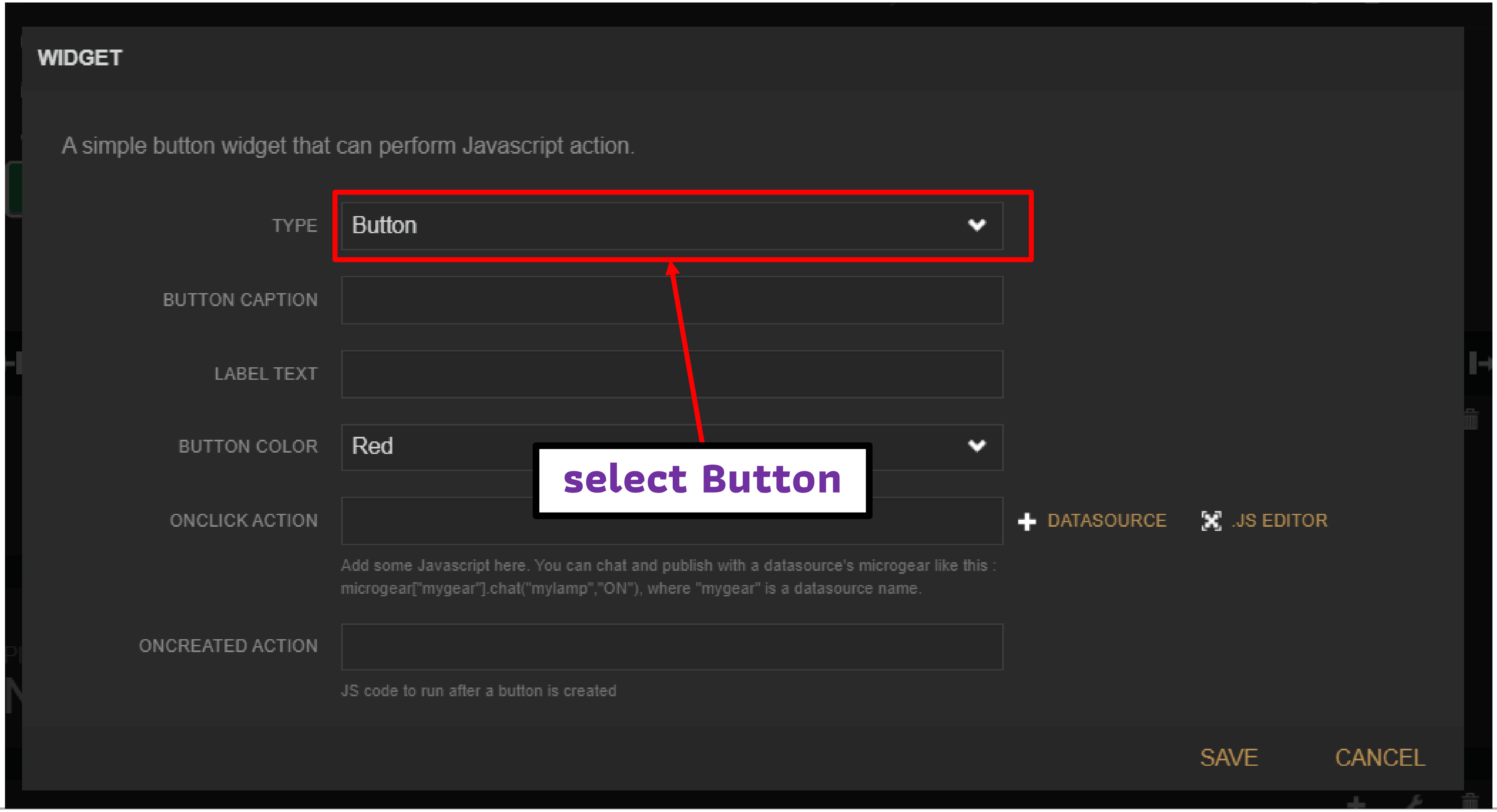
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



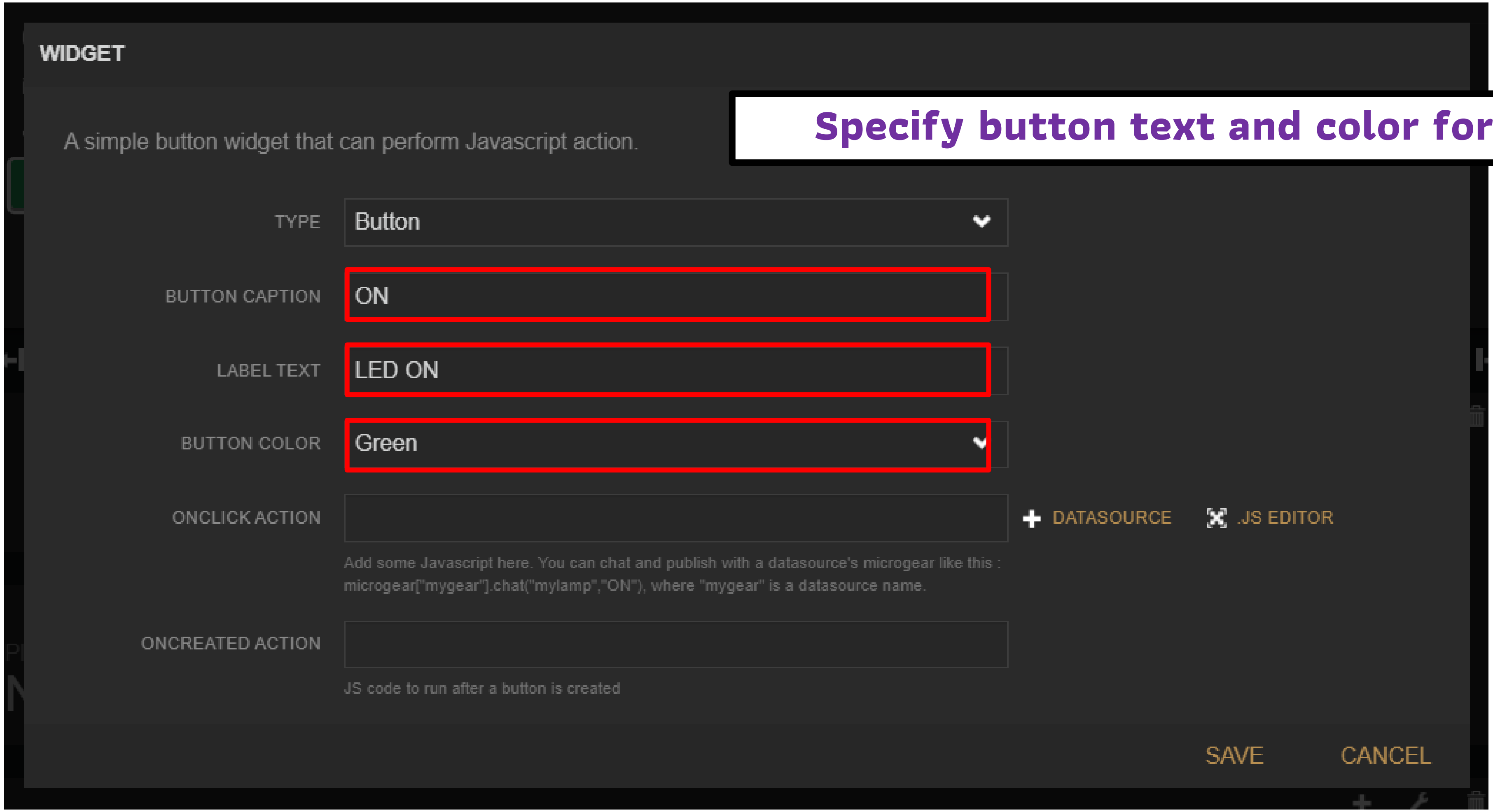
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



3 - Freeboard in NETPIE2020 (Control Widget)

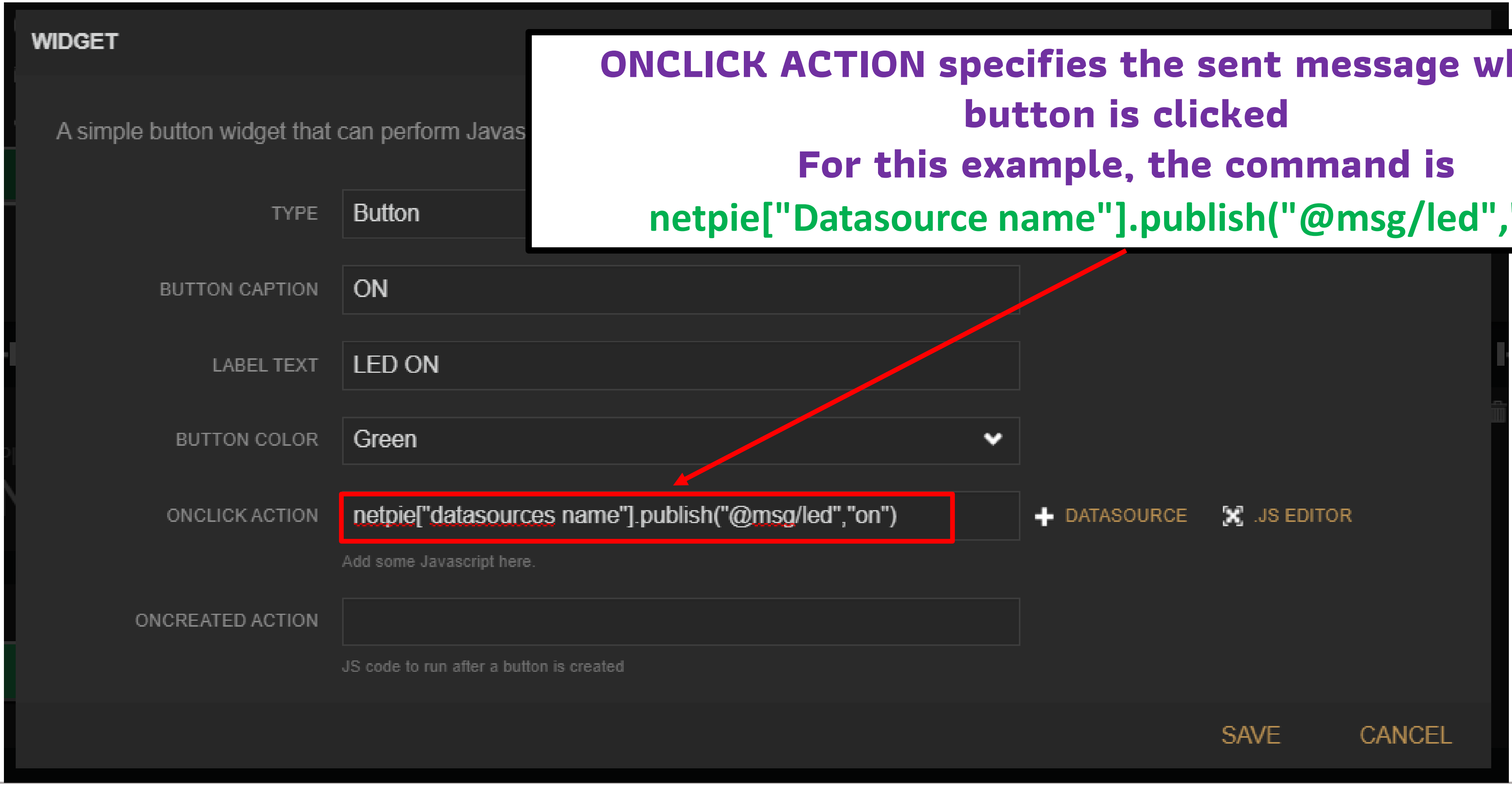
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



Specify button text and color for LED on

3 - Freeboard in NETPIE2020 (Control Widget)

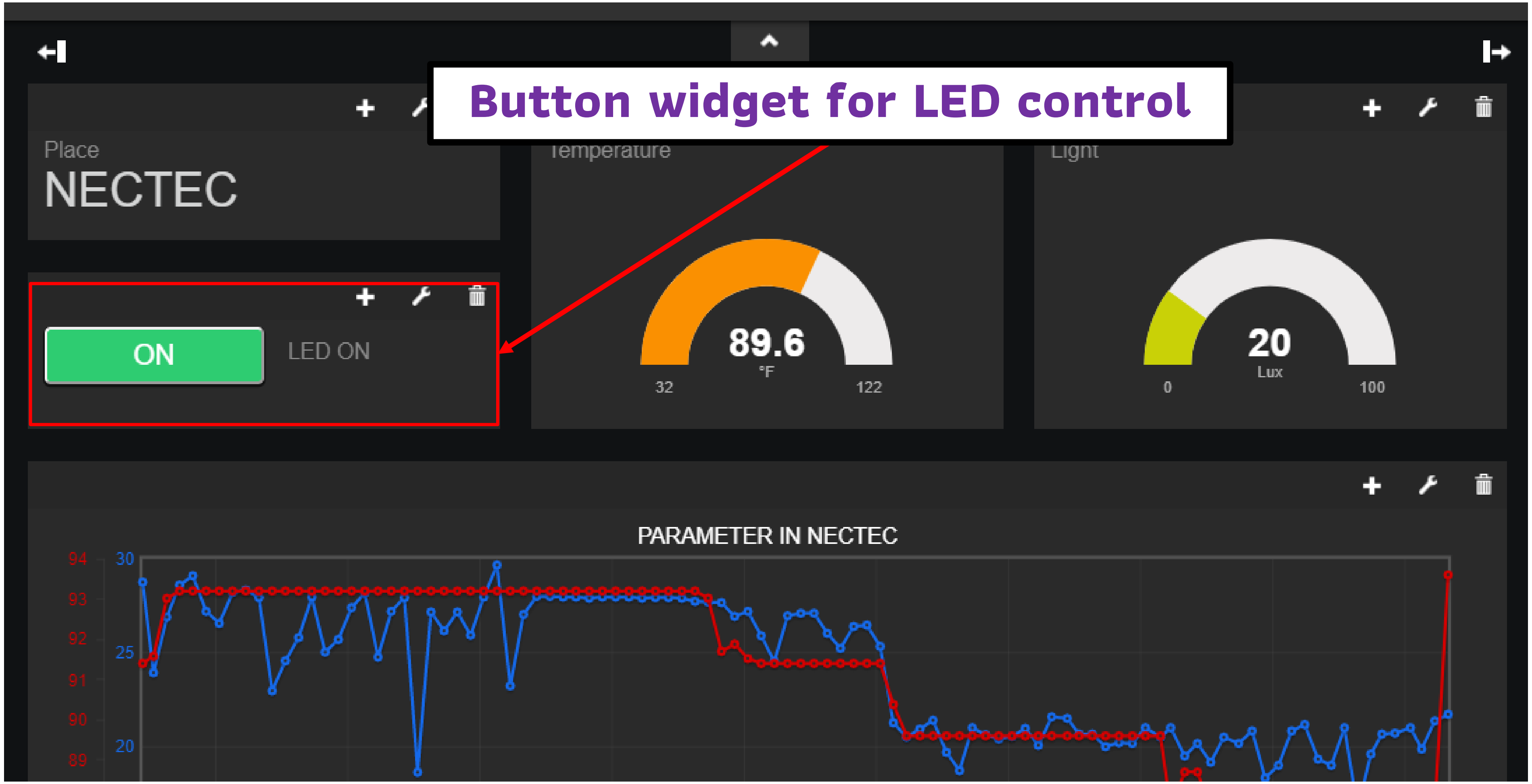
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



ONCLICK ACTION specifies the sent message when the button is clicked
For this example, the command is
netpie["Datasource name"].publish("@msg/led","on")

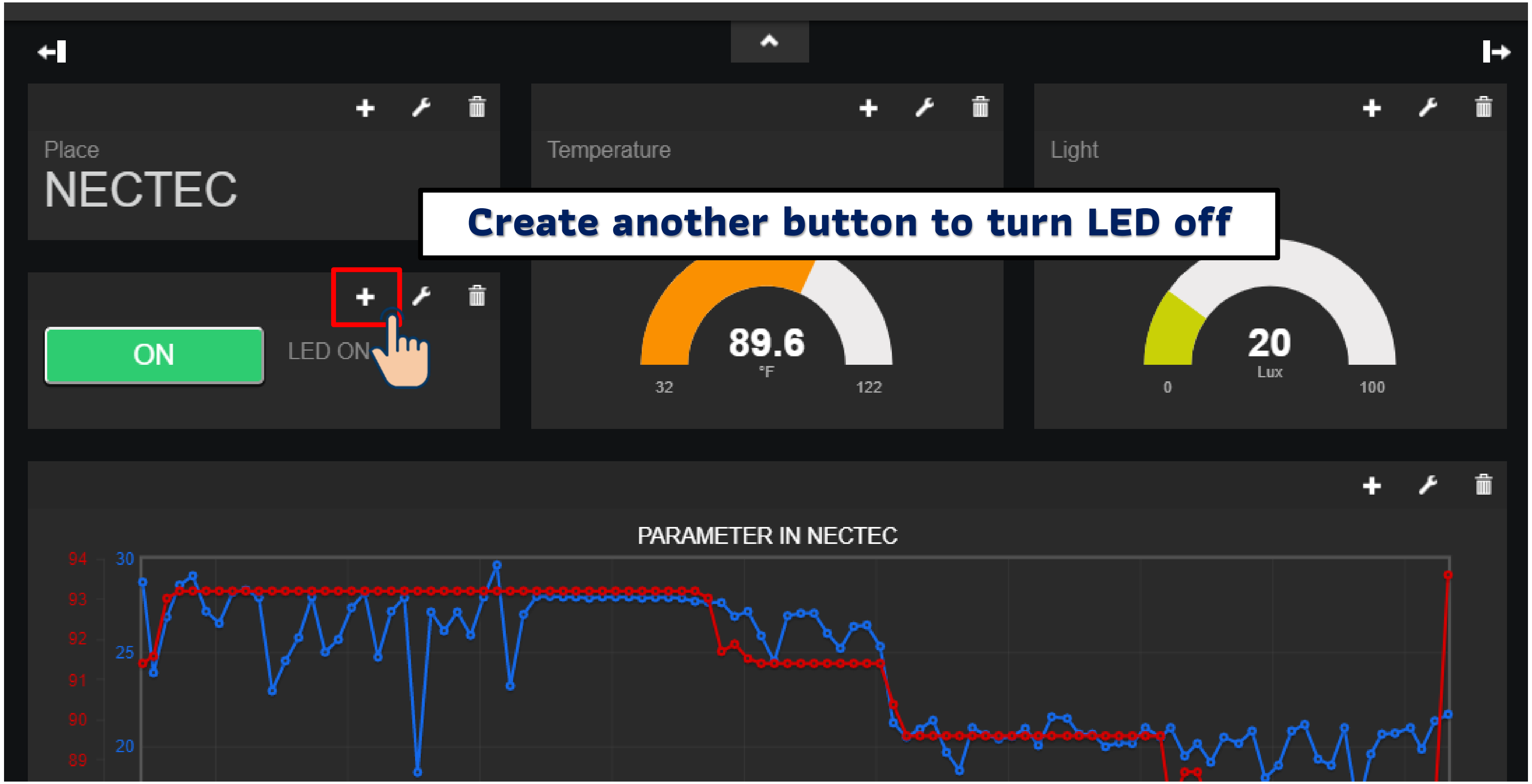
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



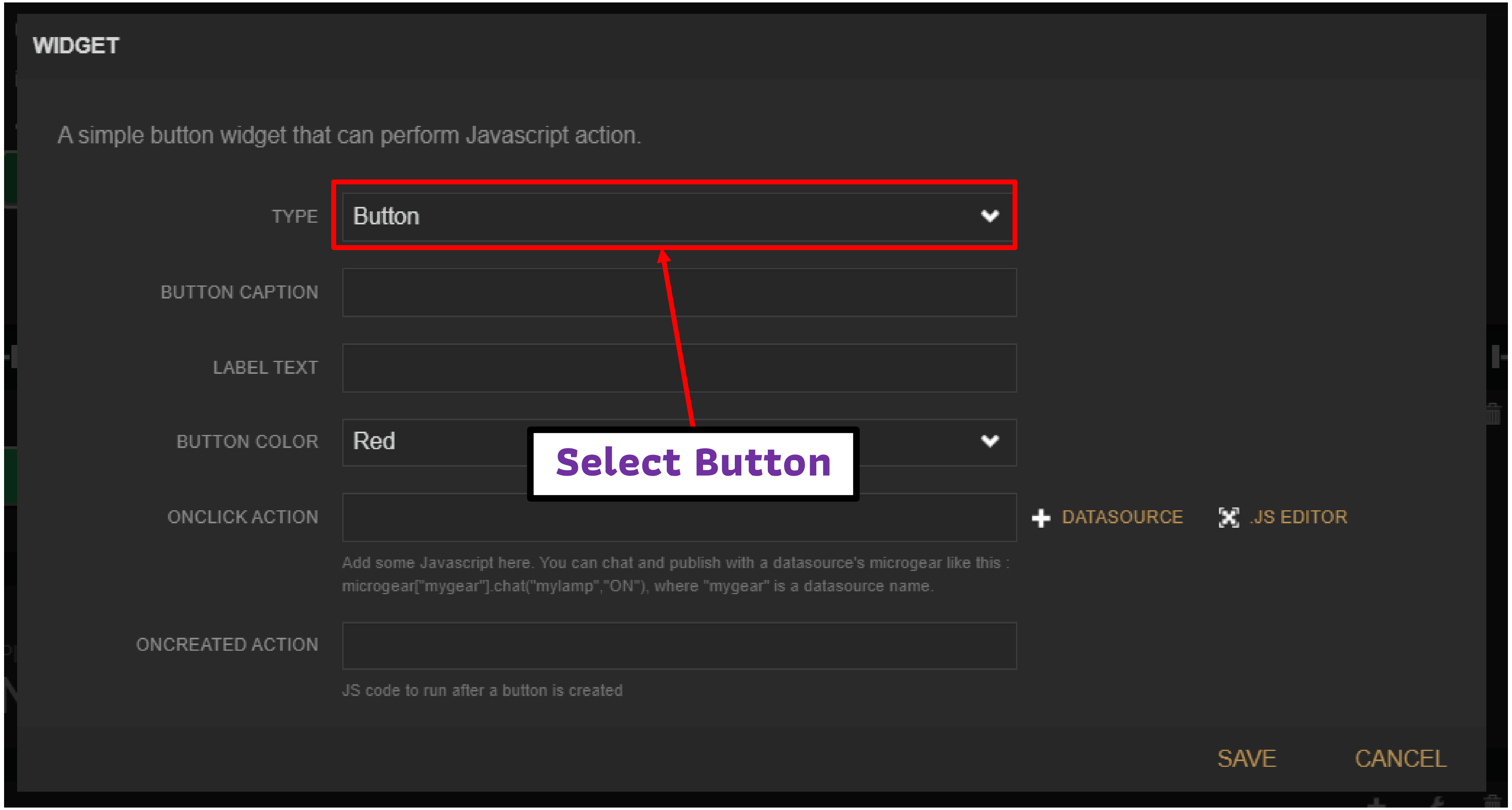
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



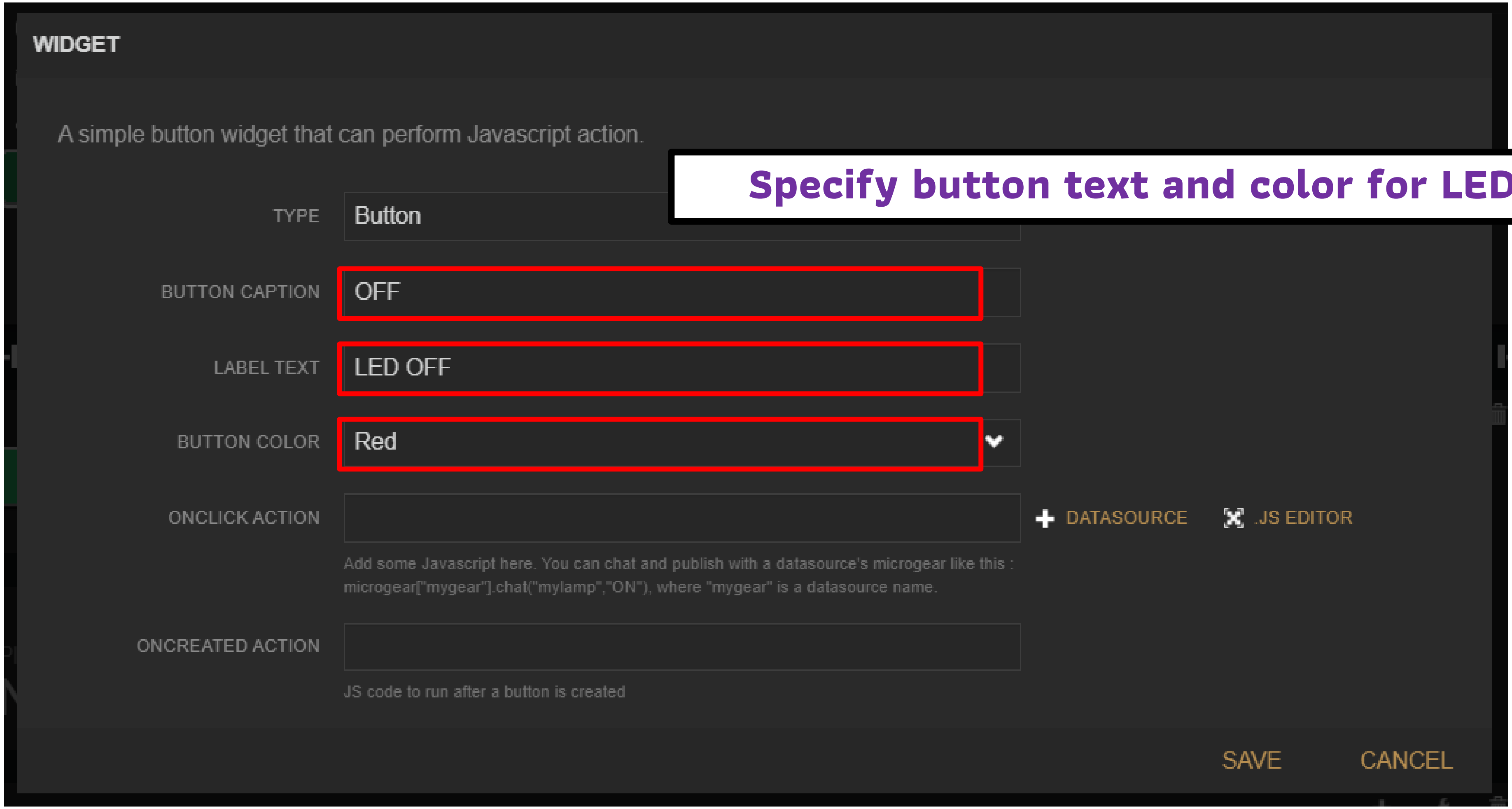
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



3 - Freeboard in NETPIE2020 (Control Widget)

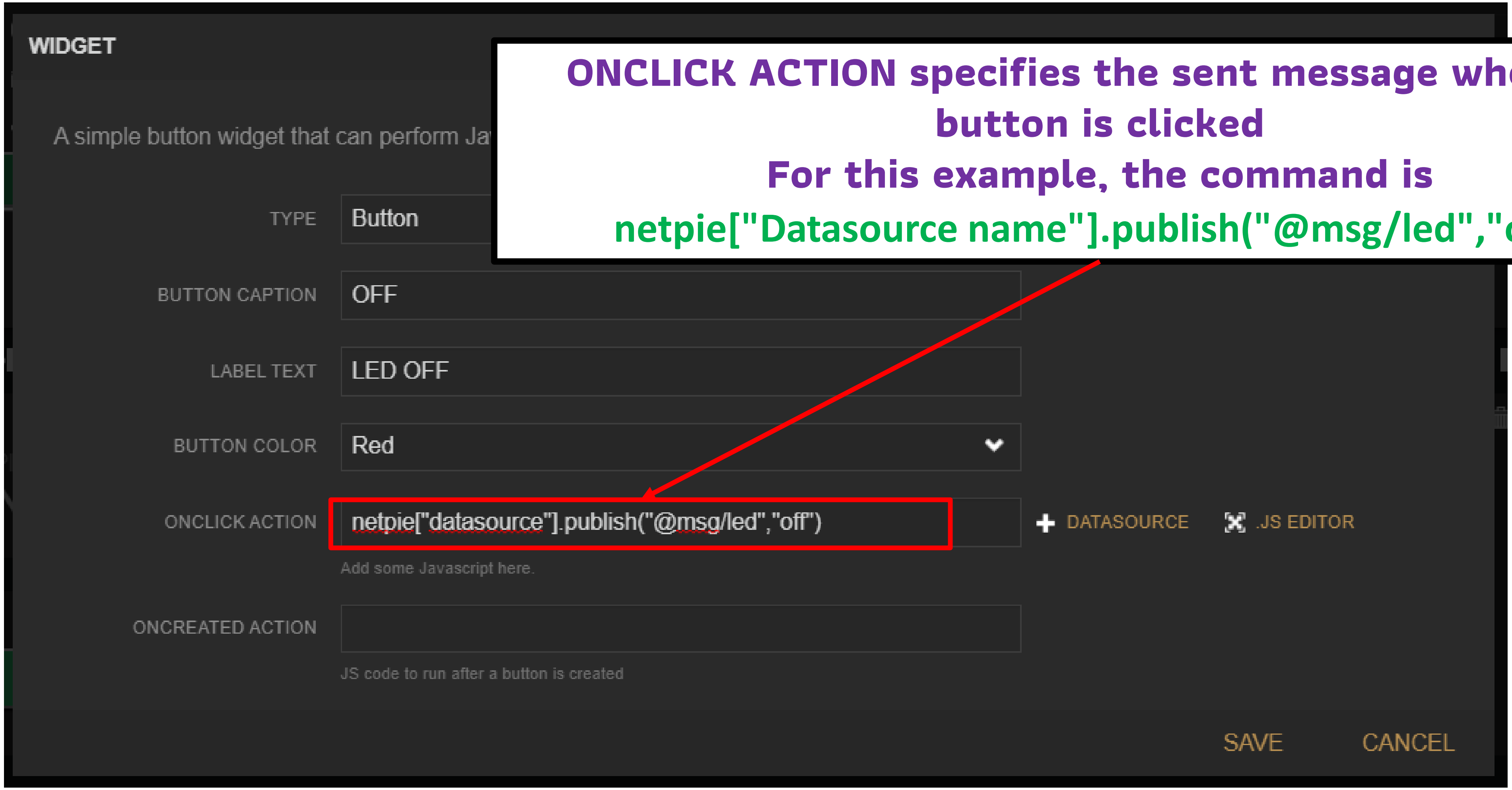
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



Specify button text and color for LED off

3 - Freeboard in NETPIE2020 (Control Widget)

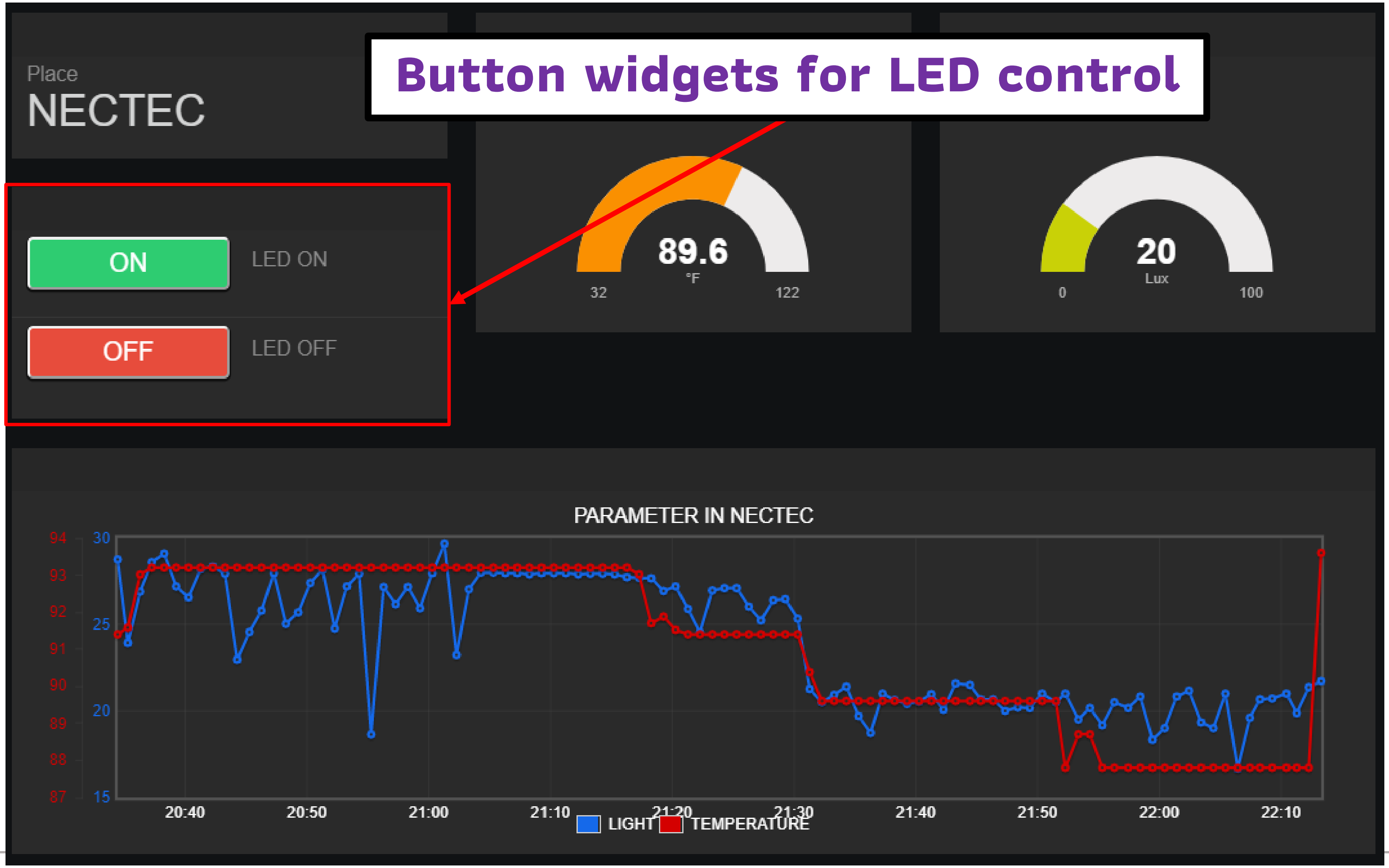
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



ONCLICK ACTION specifies the sent message when the button is clicked
For this example, the command is
netpie["Datasource name"].publish("@msg/led","off")

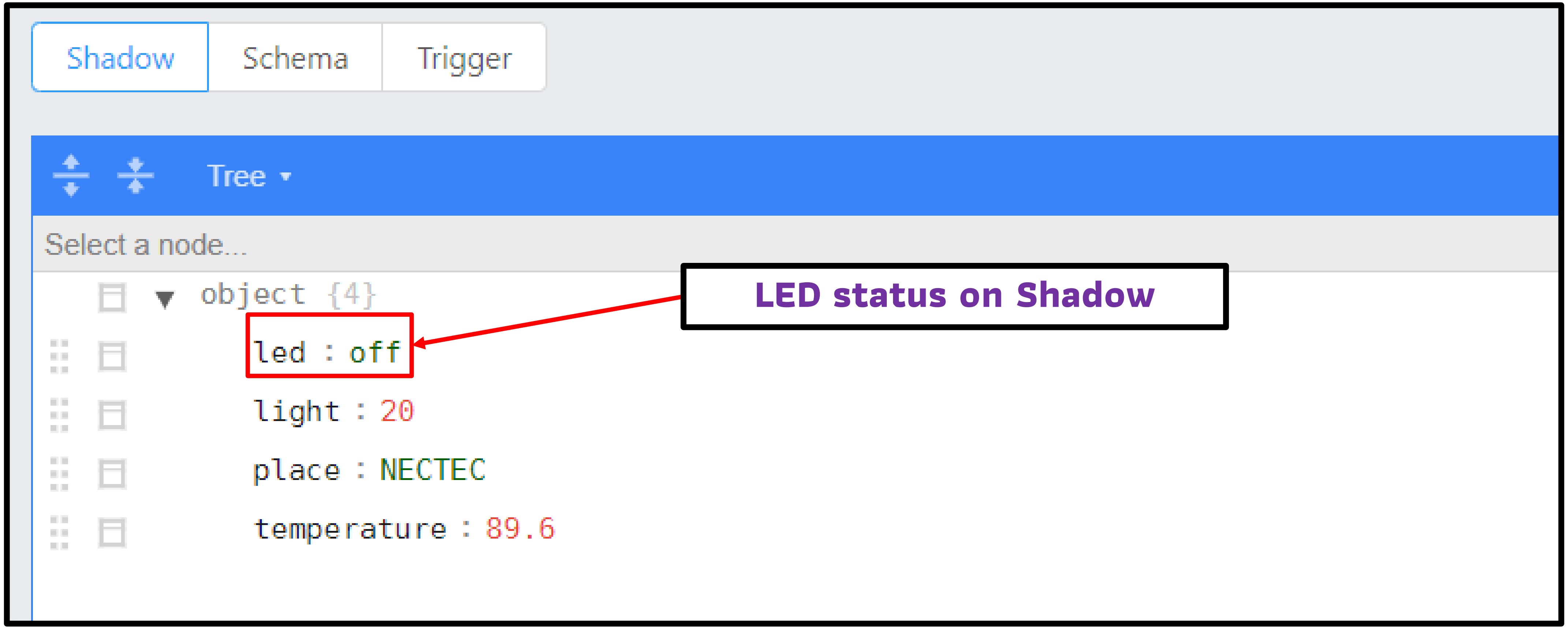
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



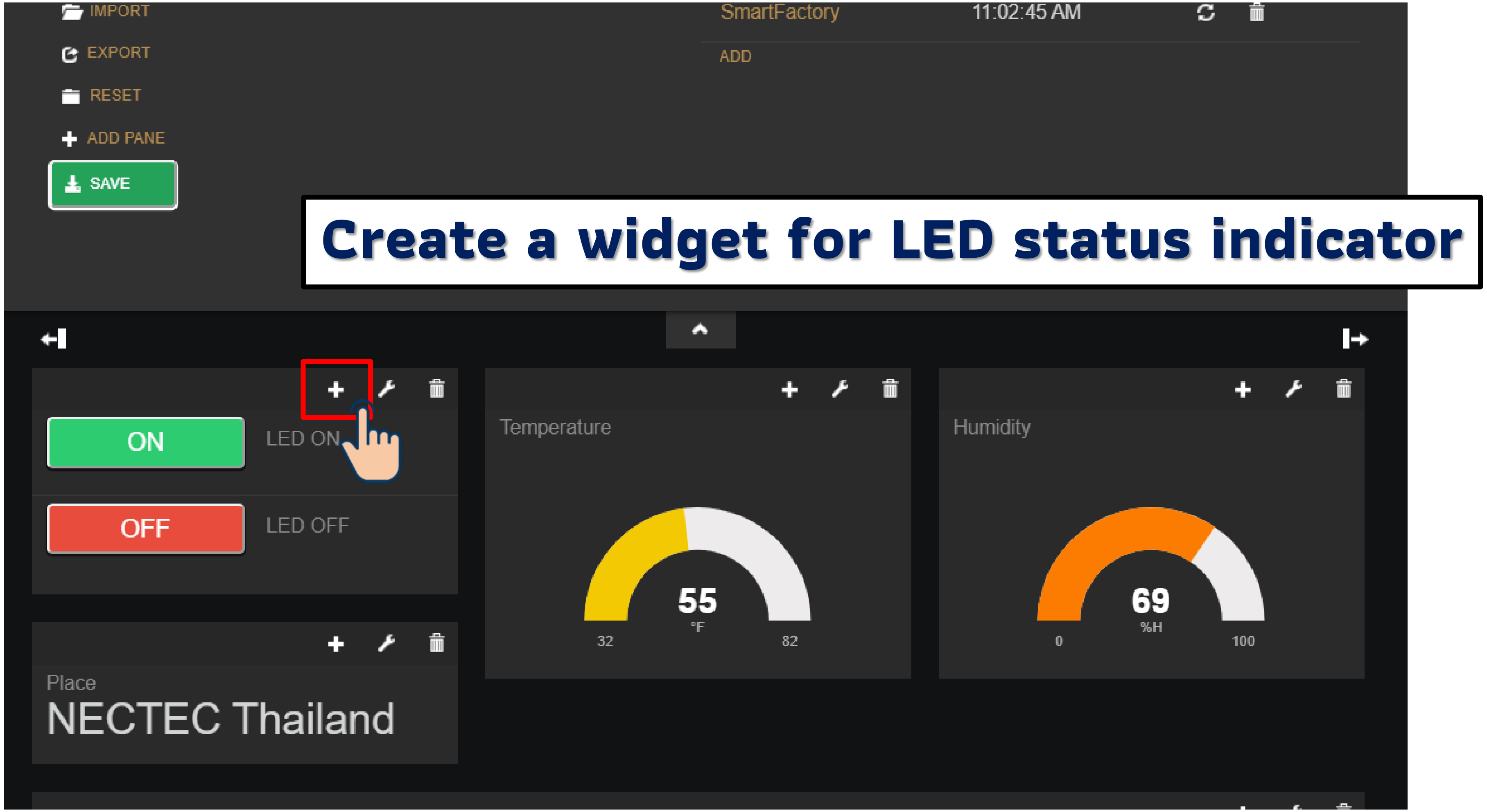
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



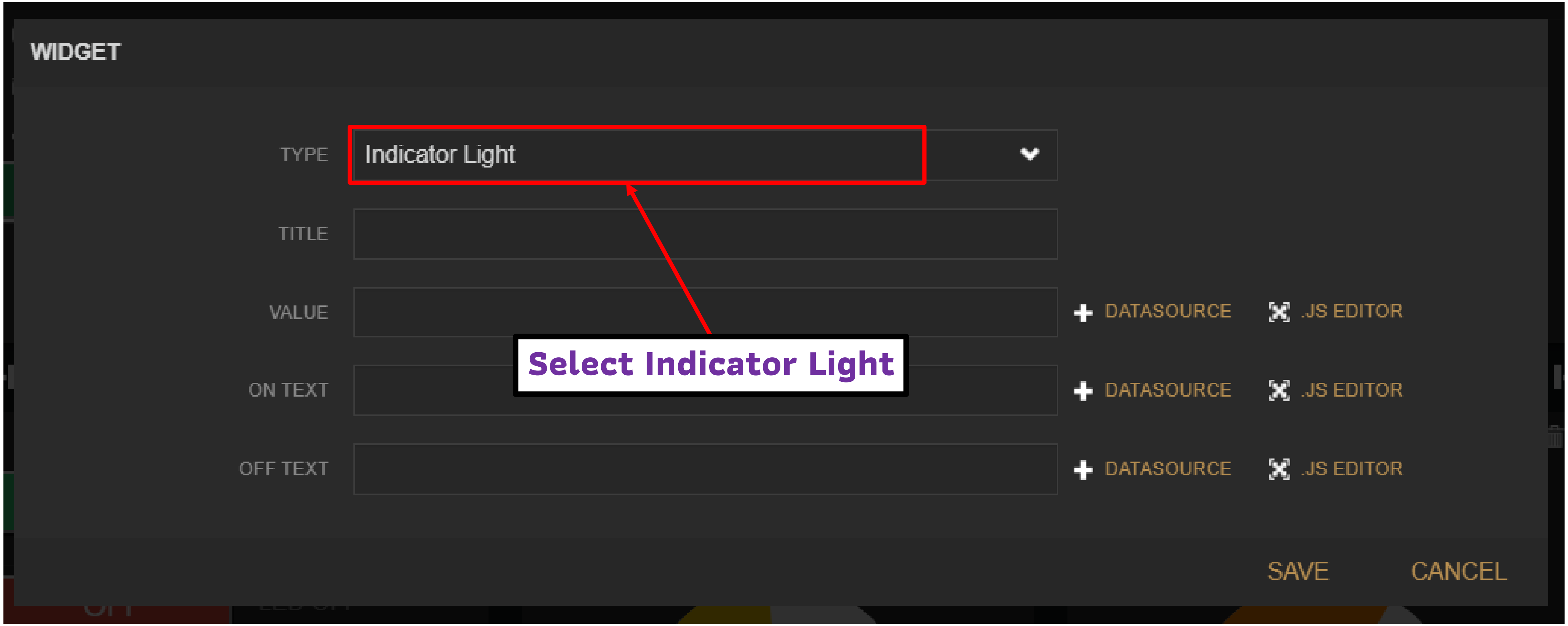
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



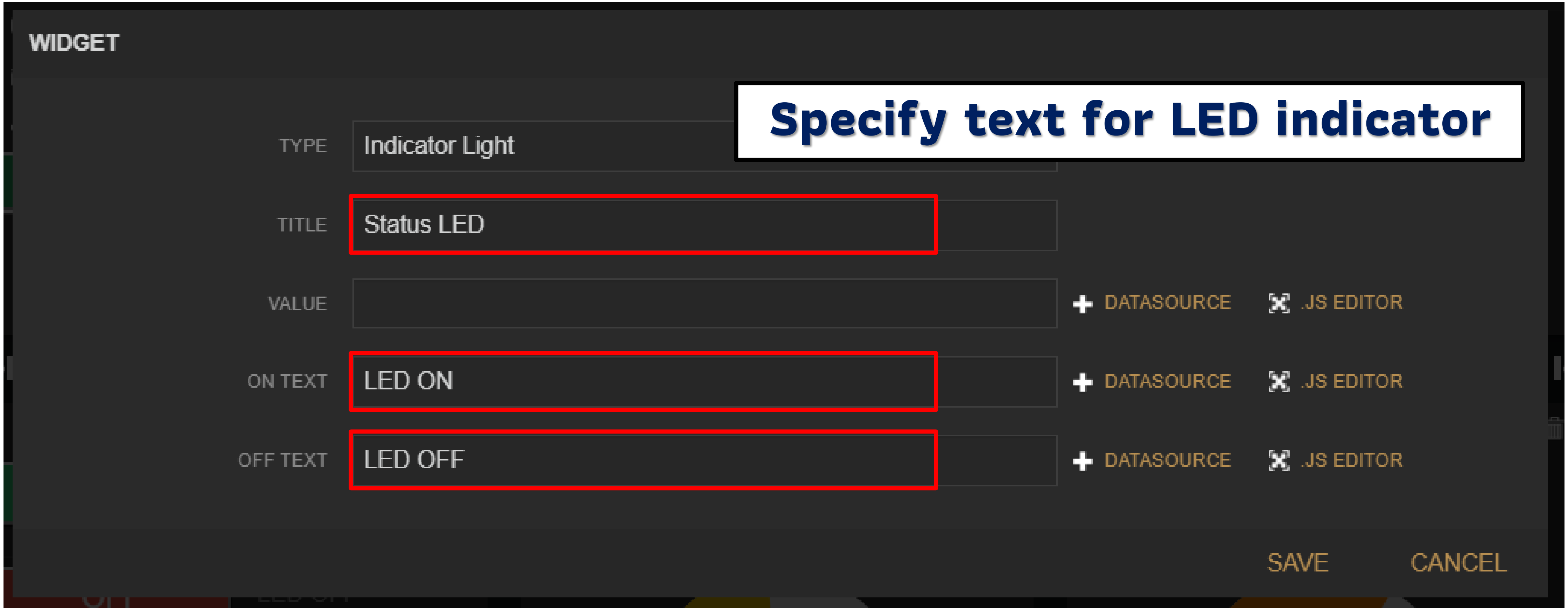
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



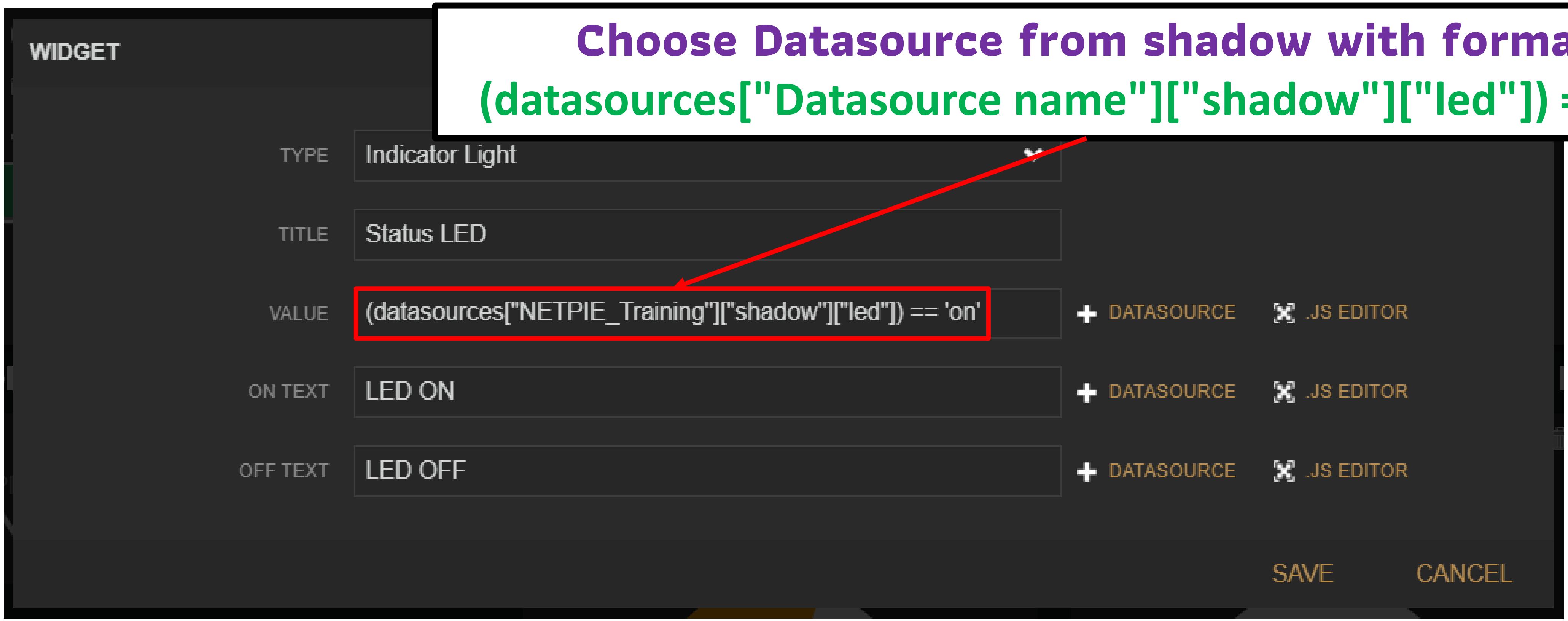
3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



3 - Freeboard in NETPIE2020 (Control Widget)

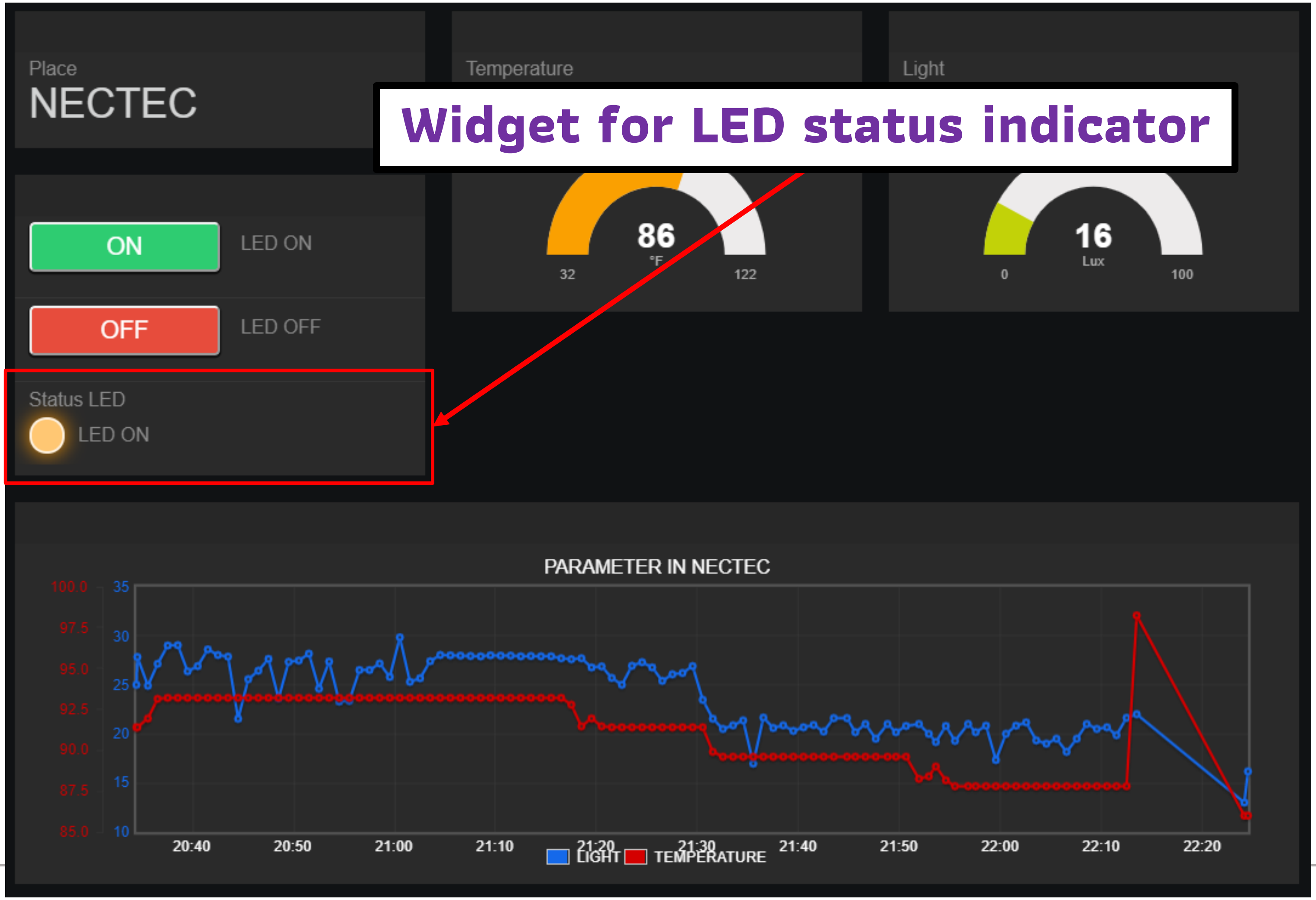
Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



Choose Datasource from shadow with format
(datasources["Datasource name"]["shadow"]["led"]) == 'on'

3 - Freeboard in NETPIE2020 (Control Widget)

Exercise 5 : Control LED on ESP32 or ESP8266 via Freeboard



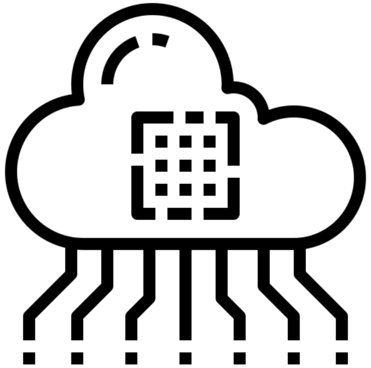
4 RESTful API

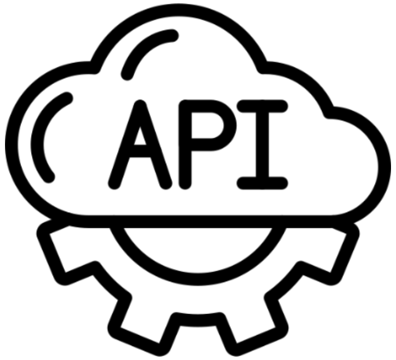
4 - RESTful API

What is RESTful API?

It is a channel for the device to call the Platform service via the RESTful API which uses HTTP protocol, suitable for integrating various systems, either existing or new development, without limitation by programming language or hardware. There are two groups of APIs:

1. Device API
2. Data Store API





4 - RESTful API

Device API

It is an API related to device management or calling. The EndPoint is https://api.netpie.io/v2/device

Publish message to a topic

```
curl -X PUT "https://api.netpie.io/v2/device/message?topic=%40msg%2Ftest" -H "Authorization: Basic ClientID:Token" -H "Content-Type: text/plain" -d "message"
```

Read Shadow Data of Device

```
curl -X GET "https://api.netpie.io/v2/device/shadow/data" -H "Authorization: Basic ClientID:Token"
```

Write Shadow Data using Merge method

```
curl -X PUT "https://api.netpie.io/v2/device/shadow/data" -d "{data:{humid:63.7, temp:25.2}}" –H "Authorization: Device ClientID:Token"
```

4 - RESTful API

Device API

Write Shadow Data using Overwrite method

```
curl -X POST "https://api.netpie.io/v2/shadow/data" -d "{data:{temp:31.7}}" -H
"Authorization: Device ClientID:Token"
```

Write to Shadow with specified time in the past

```
curl -X PUT " https://api.netpie.io/v2/shadow/data" -d "{data:{humid:61.9,
temp:28.6,timestamp:1566863843262}}" -H "Authorization: Device ClientID:Token"
```

4 - RESTful API

Data Store API

It is an API that deals with retrieving data stored in Time series Database, where the Domain name of API is
https//api.netpie.io/v2/feed . The databse is stored in KairosDB format, so querying the parameters can be done using the same format as KairosDB

Read data from Timeseries Database of Device

```
curl -X POST "https://api.netpie.io/v2/feed/api/v1/datapoints/query" –H "Content-Type: application/json" -H "Authorization: BeareruserToken" -d ' {
{\"start_relative\":{\"value\":1,\"unit\":\"days\"},\"metrics\":[{\"name\":\"ClientID\",\"tags\":{\"attr\":[\"temperature\"]},\"limit\":50,\"group_by\":[{\"name\":\"tag\",\"tags\":[\"attr\"]}],\"aggregators\":[{\"name\":\"avg\",\"sampling\":{\"value\":1,\"unit\":\"minutes\"}}]}]}"
```

Webpage to Generate Code for API calls
https://trial-api.netpie.io/

5 Workshop

NETPIE
where things chat

Drop your workshop at

Link --> https://bit.ly/3m4isJj

Set File Name is W follow by your IDStudent

Such as W3_57364150, W4_57364150

(W3 = Schema File, W4 = Freeboard File, W5 = Arduino File, W6 = Trigger File, W7 = Event Hook)

NETPIE
where things chat

NSTDA
Anniversary

NECTEC
a member of NSTDA

การใช้งาน Trigger และ Event Hook

ปิยวัฒน์ จอมสถาน
ทีมระบบไซเบอร์ – กายภาพ (CPS)
กลุ่มวิจัยไอโอทีและระบบอัตโนมัติสำหรับงานอุตสาหกรรม (IIARG)
ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ

การสร้างการแจ้งเตือนผ่าน Line Notify

LINE Notify
Connect Everything

NETPIE
where things chat

การสร้างการแจ้งเตือนผ่าน Line Notify

Device Trigger and Event Hook

Device Trigger

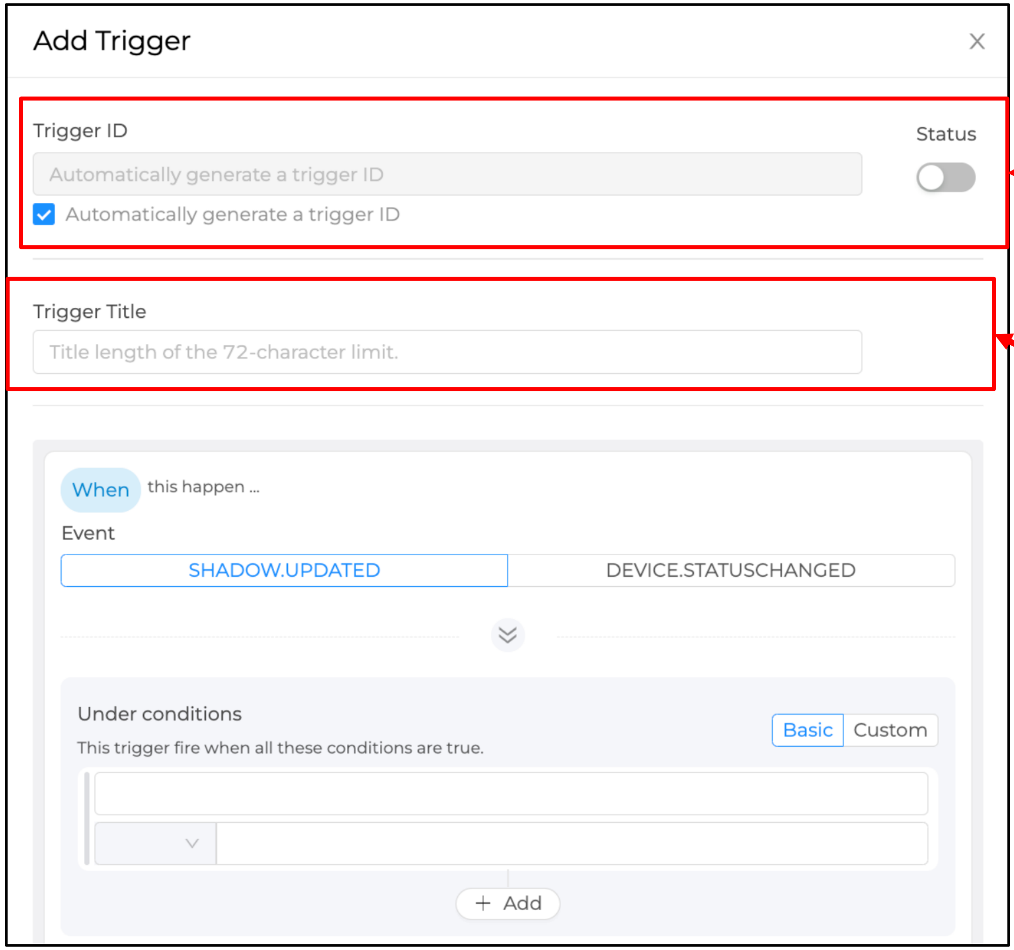
เป็นระบบที่ผูกกับการเปลี่ยนแปลงข้อมูลของ Device Shadow เข้ากับการกระทำภายนอก (Event Hook) เช่น การตั้งค่าแจ้งเตือนตามสถานะต่างๆ ตามเงื่อนไขการทำงานของ Device ที่ถูกตั้งค่าไว้

Event Hook

เป็นตัวกลางที่ใช้กำหนดว่าเมื่อเกิด Trigger จะให้ดำเนินการอะไร ผ่าน 3rd Party หรือระบบอื่นๆที่ต้องการใช้ เช่น การแจ้งเตือนตามสถานะต่างๆผ่าน Line Notify

การสร้างการแจ้งเตือนผ่าน Line Notify

ส่วนต่างๆของ Device Trigger



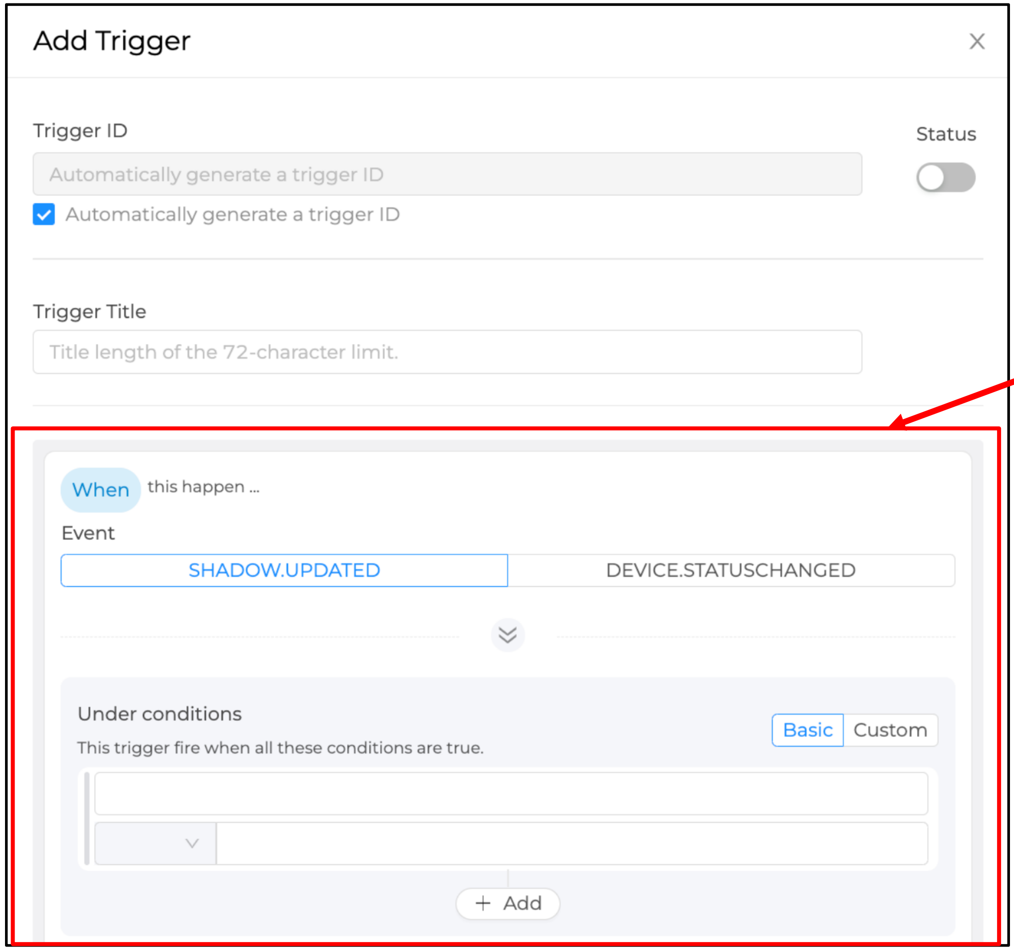
Trigger ID (string) : รหัสของ Trigger ซึ่งระบบจะสร้างให้อัตโนมัติหรือผู้ใช้ต้องการกำหนดเองก็ได้

Status : สถานะเปิด/ปิดการใช้งาน Trigger

Trigger Title (string) : ชื่อหรือคำอธิบายสั้น ๆ เกี่ยวกับ Trigger

การสร้างการแจ้งเตือนผ่าน Line Notify

ส่วนต่างๆของ Device Trigger



Event : ประเภทการเปลี่ยนแปลงข้อมูลของ Device (Device Shadow) มี 2 ประเภท คือ

1. SHADOW.UPDATED จะเกิด Trigger เมื่อ Device Shadow Data มีการเปลี่ยนแปลงตรงตามเงื่อนไข (Under conditions) ที่กำหนดไว้

2. DEVICE.STATUSCHANGED จะเกิด Trigger เมื่อ Device เปลี่ยนสถานะการเชื่อมต่อ Platform (Online/Offline) และตรงตามเงื่อนไข (Under conditions) ที่กำหนดไว้ ซึ่งการกำหนดเงื่อนไขสำหรับ Trigger Event นี้มีได้ 3 รูปแบบ ดังนี้

   1. ต้องการให้ Trigger ทุกครั้งที่สถานะการเชื่อมต่อ Platform เปลี่ยนไม่ว่าจะ Online เป็น Offline หรือ Offline เป็น Online ให้เซ็ตเงื่อนไข (Under conditions) ให้เป็นจริงเสมอ เช่น true == true หรือ 1 == 1 เป็นต้น

   2. ต้องการให้ Trigger ในกรณีที่เปลี่ยนสถานะเป็น Online เท่านั้น ให้เซ็ต ให้เซ็ตเงื่อนไข (Under conditions) เป็น $NEW.STATUS == 1

   3. ต้องการให้ Trigger ในกรณีที่เปลี่ยนสถานะเป็น Offline เท่านั้น ให้เซ็ต ให้เซ็ตเงื่อนไข (Under conditions) เป็น $NEW.STATUS == 0

การสร้างการแจ้งเตือนผ่าน Line Notify

ส่วนต่างๆของ Device Trigger

Action to : เลือก Event hook ที่ต้องการให้ทำงานต่อเมื่อเกิดการ Trigger โดยรายการใน Dropdown จะได้มาจากการรายการที่ถูกสร้างในเมนู Event Hooks ด้านซ้ายมือ

With this context variables : ประกาศตัวแปรที่จะส่งไปเรียกใช้ใน Event hook โดยทำการประกาศชื่อตัวแปรในช่องฝั่งซ้ายมือ และกำหนดค่าในช่องฝั่งขวามือ ซึ่งค่าที่กำหนดจะเป็นค่าคงที่ ค่าตัวแปรจาก Shadow หรือค่าตัวแปรจากระบบมีให้เรียกใช้ได้ ส่วนการอ้างอิงเพื่อใช้งานที่ Event hook จะใช้เป็น {{context.ชื่อตัวแปร}}

การสร้างการแจ้งเตือนผ่าน Line Notify

การอ้างอิงค่า Shadow ใน Trigger

สำหรับการอ้างอิงค่าตัวแปร Shadow สามารถเรียกใช้ใน Condition หรือ Context Variable ของ Trigger มี 3 รูปแบบ ดังนี้

1. **$CUR.พาธของตัวแปร** คือ ค่าปัจจุบันล่าสุดที่ถูกอัพเดท ($NEW merge $PREV) โดยขึ้นต้นด้วย $CUR ตามด้วย Path ตามโครงสร้างใน Shadow
2. **$NEW.พาธของตัวแปร** ค่าใหม่ที่ส่งมาอัพเดทลง Shadow โดยขึ้นต้นด้วย $NEW ตามด้วย Path ตามโครงสร้างใน Shadow
3. **$PREV.พาธของตัวแปร** ค่าก่อนหน้าที่จะถูกอัพเดทลง Shadow โดยขึ้นต้นด้วย $PREV ตามด้วย Path ตามโครงสร้างใน Shadow

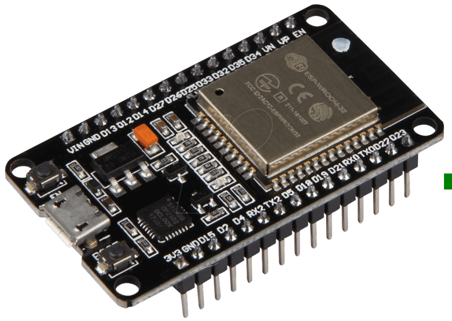
ทั้งนี้ Device Trigger มีการอ้างอิงตัวแปรอื่นๆในระบบอีกมากมาย สามารถดูรายละเอียดได้ตามนี้
https://docs.netpie.io/device-config.html#device-trigger-and-event-hook

การสร้างการแจ้งเตือนผ่าน Line Notify

ตัวอย่างการทำงานของ Trigger and Event Hook

Event Hooks

LINE NOTIFY



ESP8266/ESP32

Publish : @shadow/data/update

NETPIE2020

Device2 Shadow

"temperature" : 74

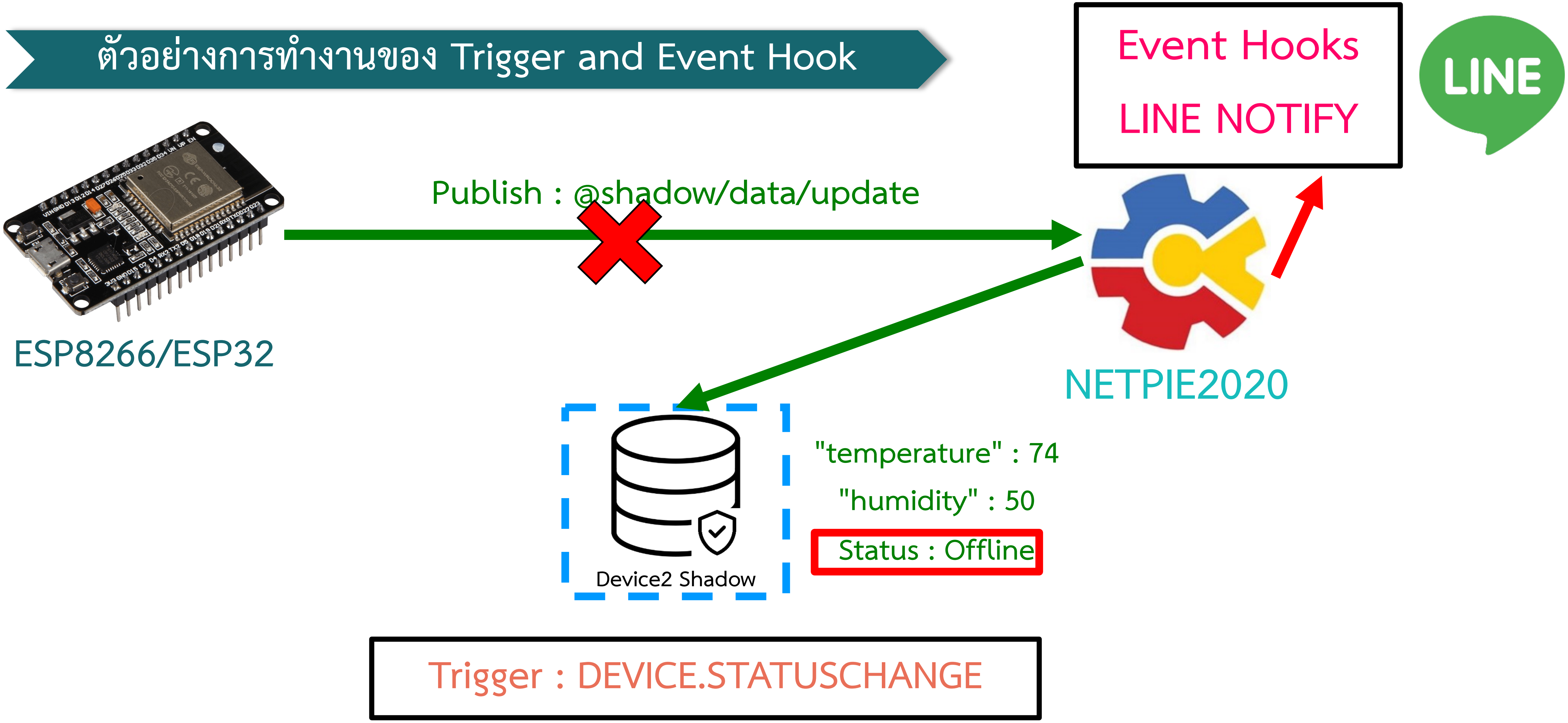
"humidity" : 50

Status : Online

Trigger : DEVICE.STATUSCHANGE

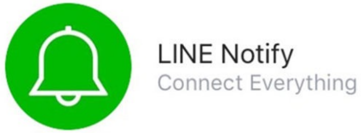
การสร้างการแจ้งเตือนผ่าน Line Notify

ตัวอย่างการทำงานของ Trigger and Event Hook

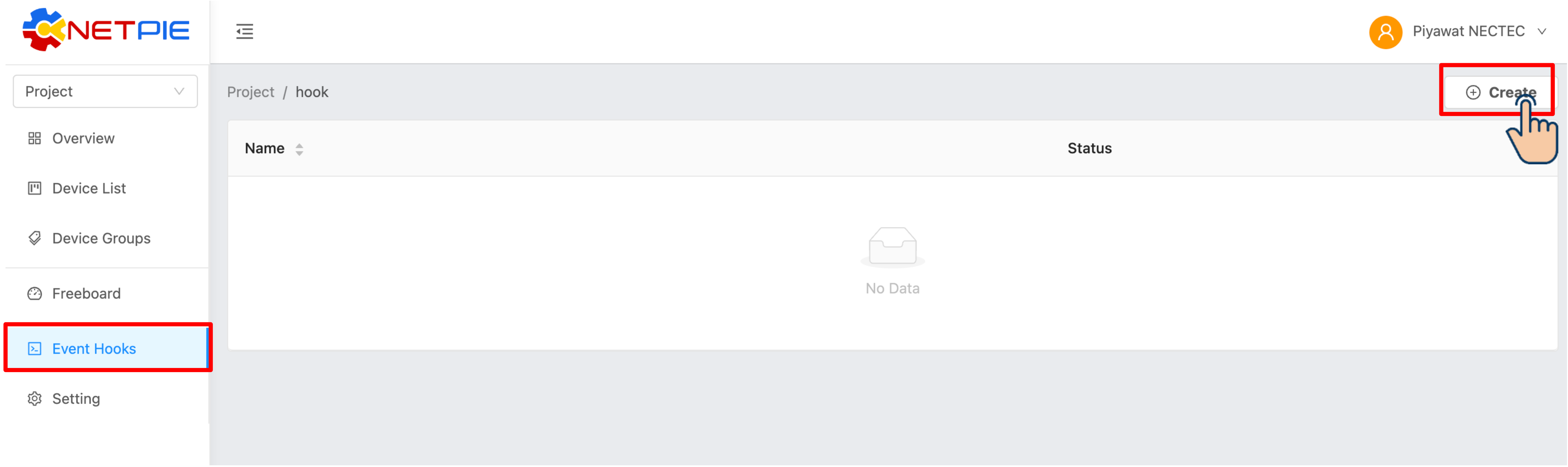


การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



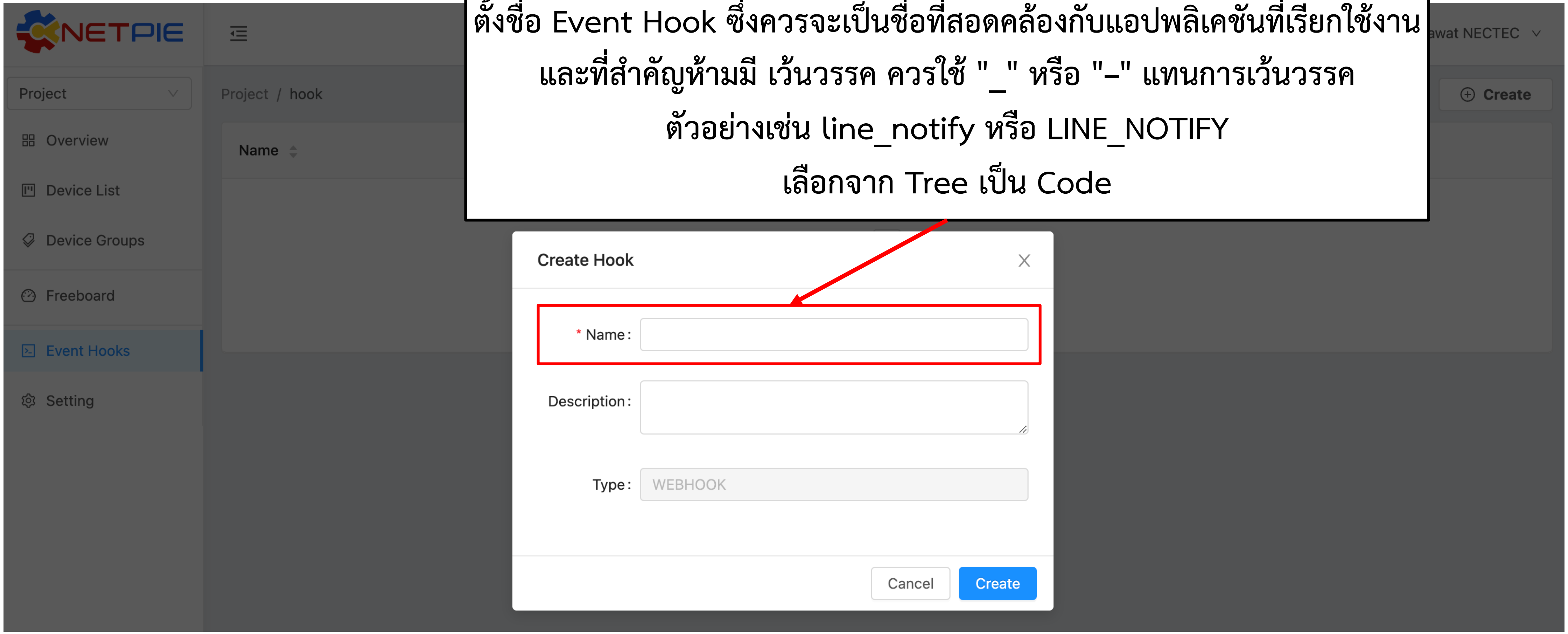
เราจะทำการสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify ซึ่งเป็น 3rd Party ที่ได้รับความนิยมสำหรับการทำระบบแจ้งเตือน



การสร้างการแจ้งเตือนผ่าน Line Notify

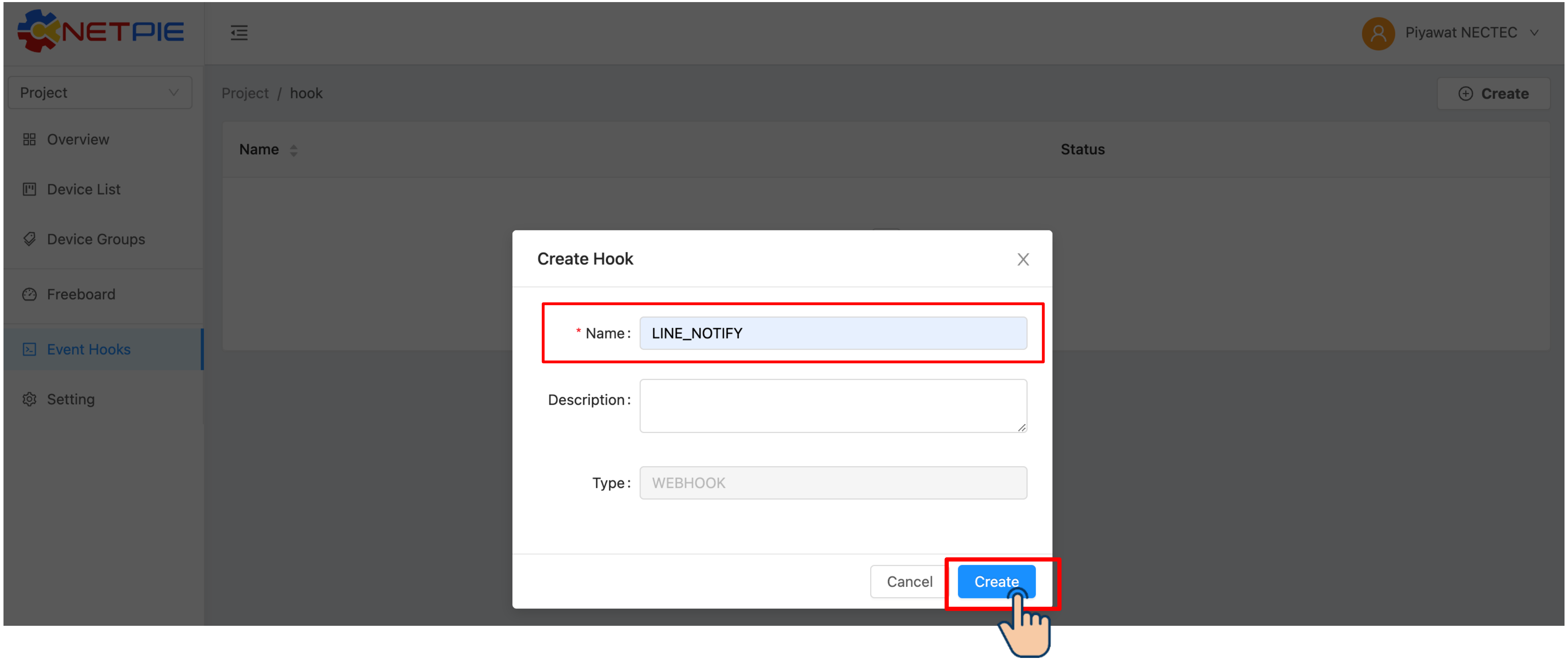
การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify

ตั้งชื่อ Event Hook ซึ่งควรจะเป็นชื่อที่สอดคล้องกับแอปพลิเคชันที่เรียกใช้งาน
และที่สำคัญห้ามมี เว้นวรรค ควรใช้ "_" หรือ "–" แทนการเว้นวรรค
ตัวอย่างเช่น line_notify หรือ LINE_NOTIFY
เลือกจาก Tree เป็น Code



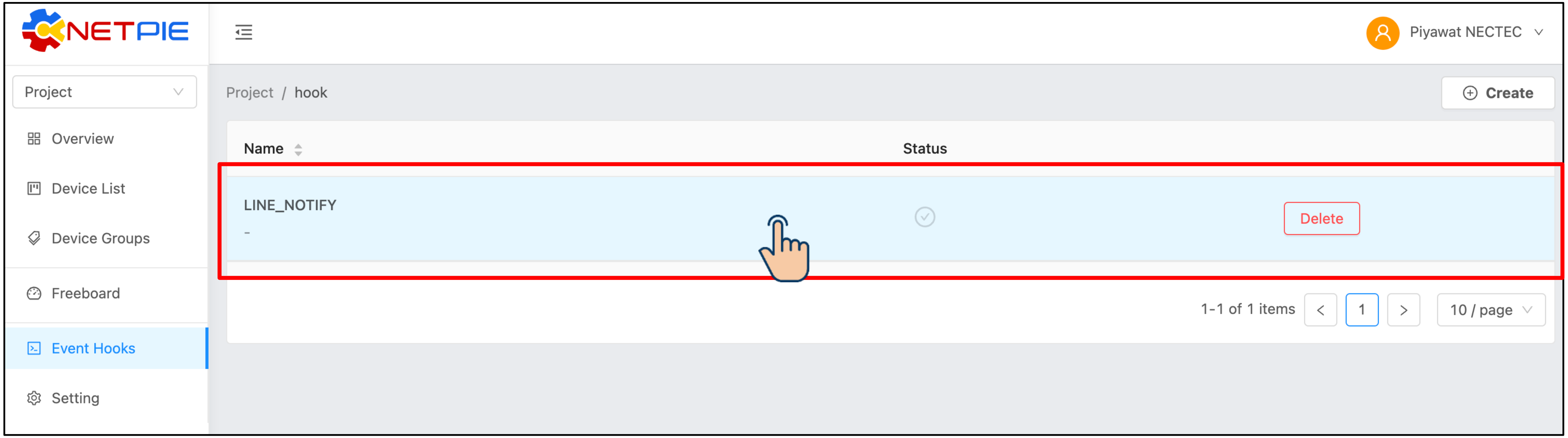
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



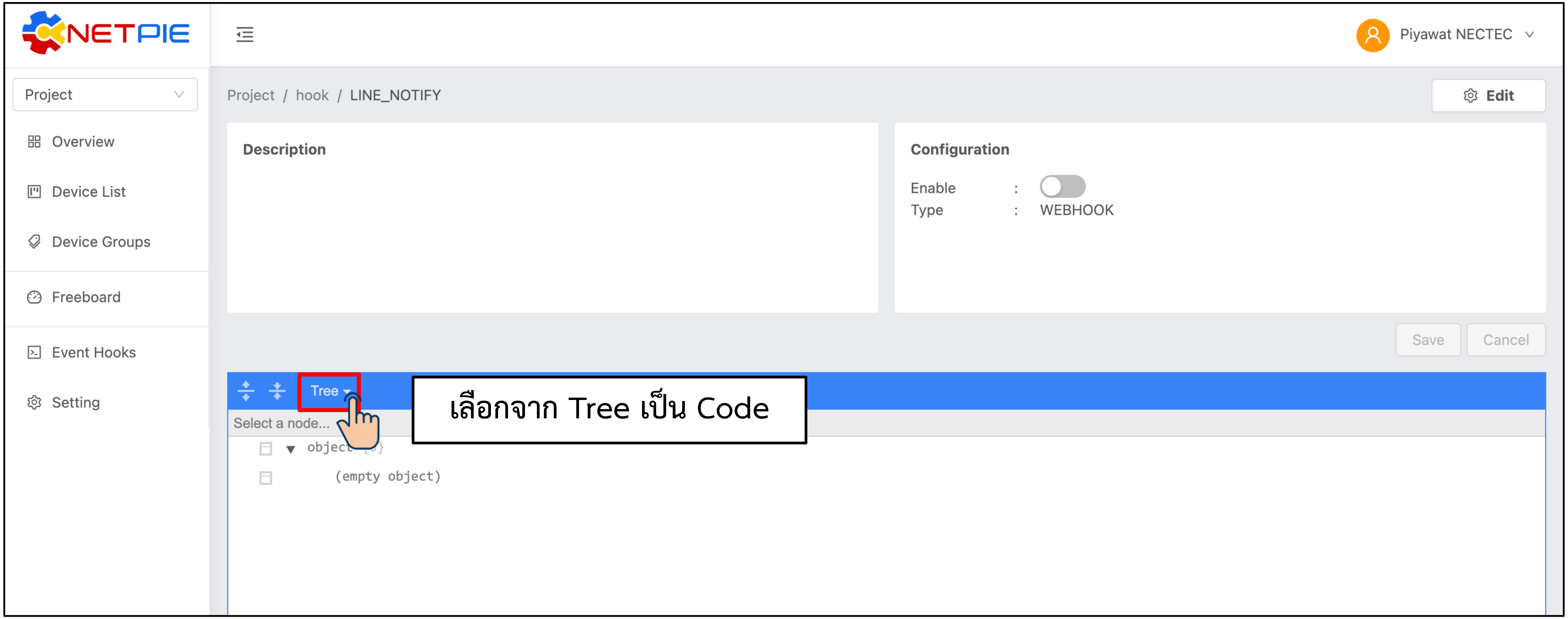
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify

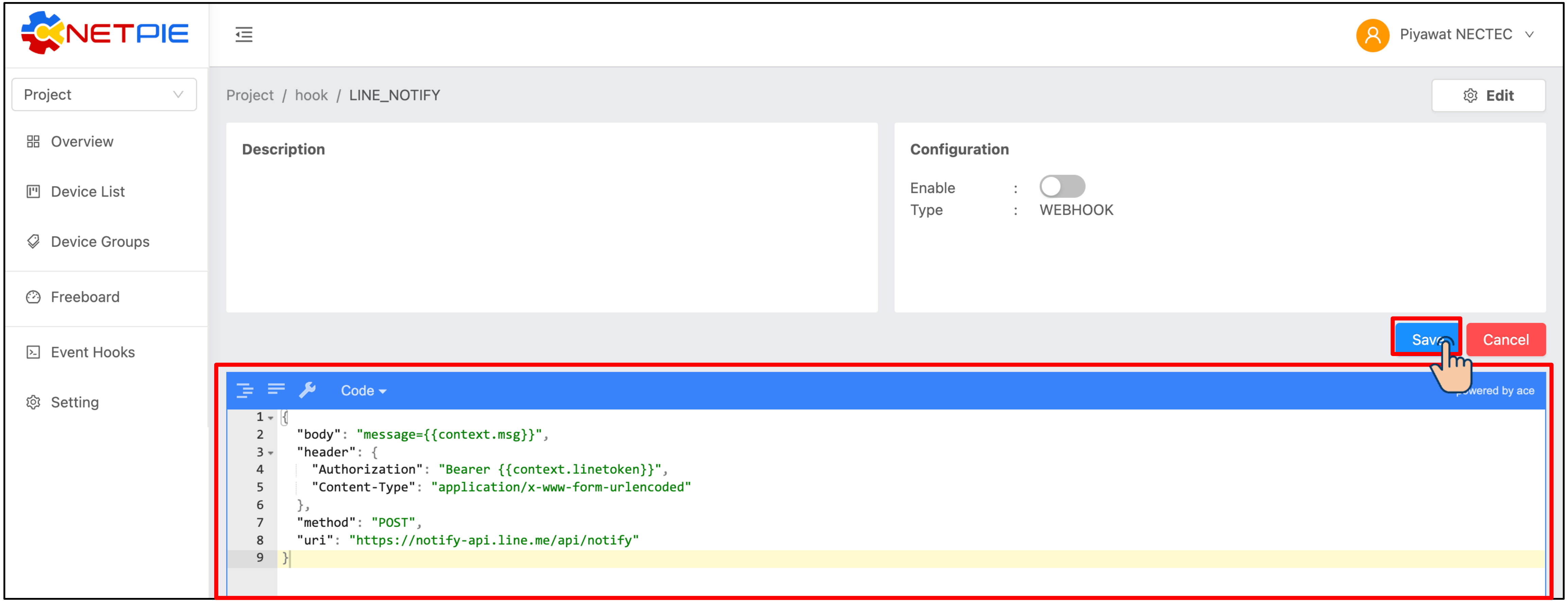
Event Hook Example เป็นตัวอย่างการทำ Line Notify ซึ่งกำหนดค่าได้ 4 Attributes

```
{
  "body": "message={{context.msg}}",
  "header": {
    "Authorization": "Bearer {{context.linetoken}}",
    "Content-Type": "application/x-www-form-urlencoded"
  },
  "method": "POST",
  "uri": "https://notify-api.line.me/api/notify"
}
```

คัดลอกทั้งหมดไปไว้ใน Event Hook บน NETPIE2020

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify

คำอธิบายการใช้งาน Line Notify

```
{
  "body": "message={{context.msg}}",
  "header": {
    "Authorization": "Bearer {{context.linetoken}}",
    "Content-Type": "application/x-www-form-urlencoded"
  },
  "method": "POST",
  "uri": "https://notify-api.line.me/api/notify"
}
```

body ส่วนของข้อมูลในที่นี้คือ ข้อความที่จะส่งไปยังปลายทาง

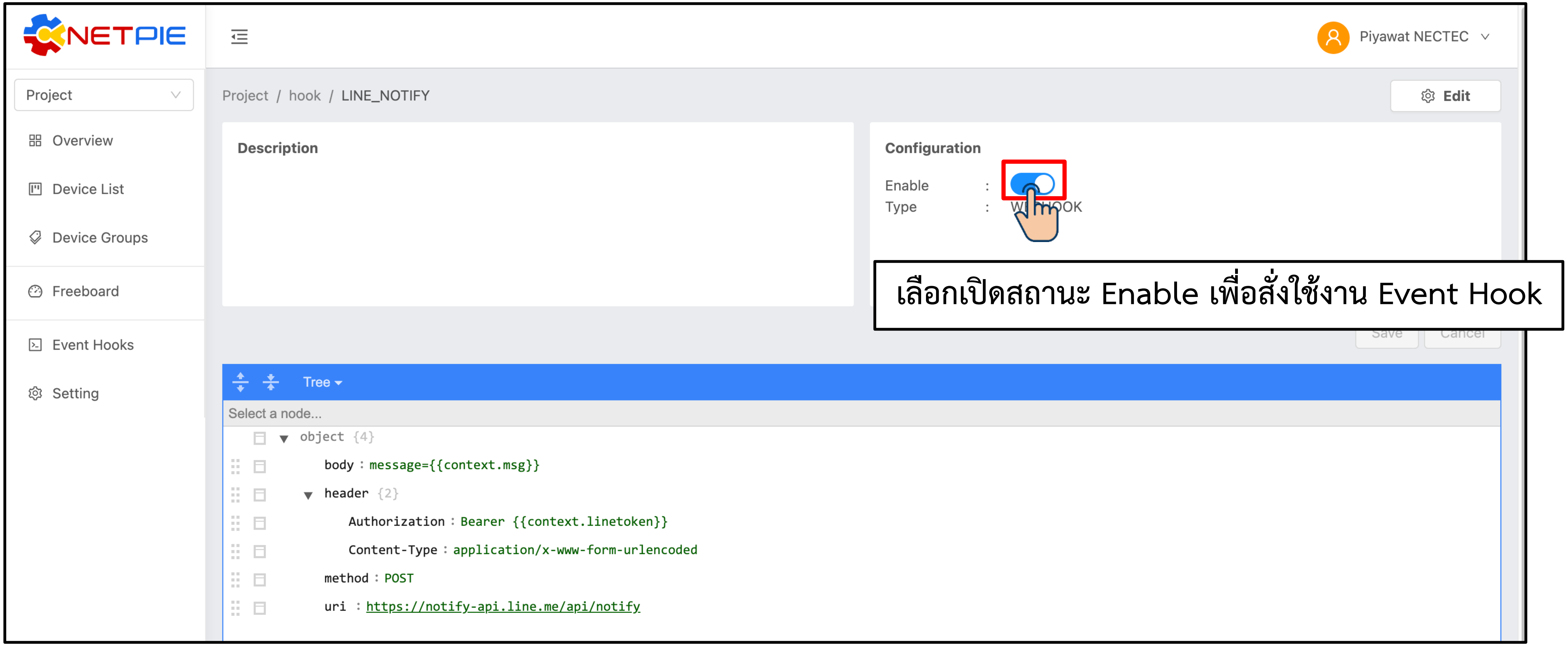
header คือ ข้อมูลเพิ่มเติมที่ต้องการส่งไปยังปลายทาง เช่น Authorization, Content-Type เป็นต้น

method คือ ส่วนที่กำหนดว่าปลายทางต้องการให้ส่งไปในแบบไหน GET, POST หรือ PUT

uri คือ Endpoint ปลายทางที่กำหนดว่าต้องการให้ส่งไปที่ใด

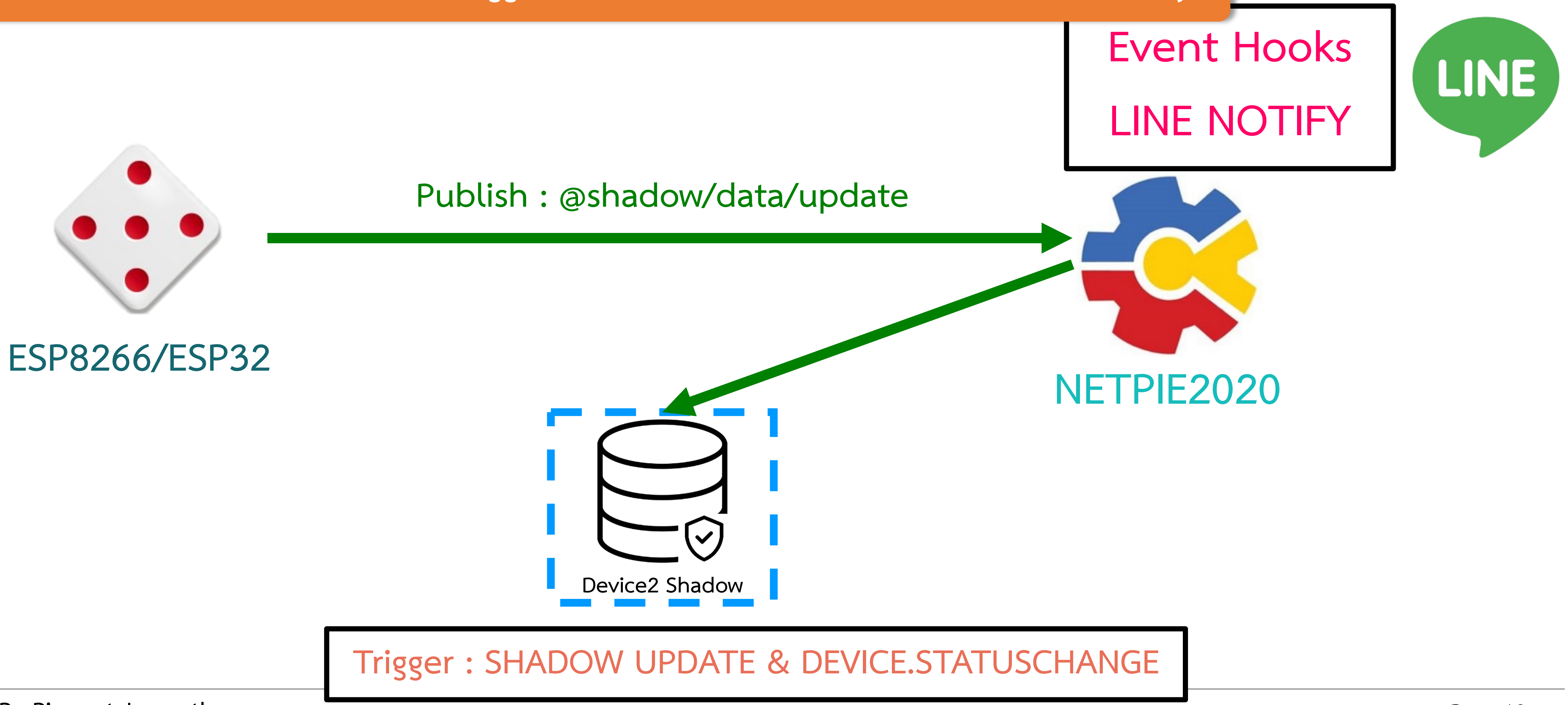
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 1 การสร้าง Event Hook สำหรับเรียกใช้งาน Line Notify



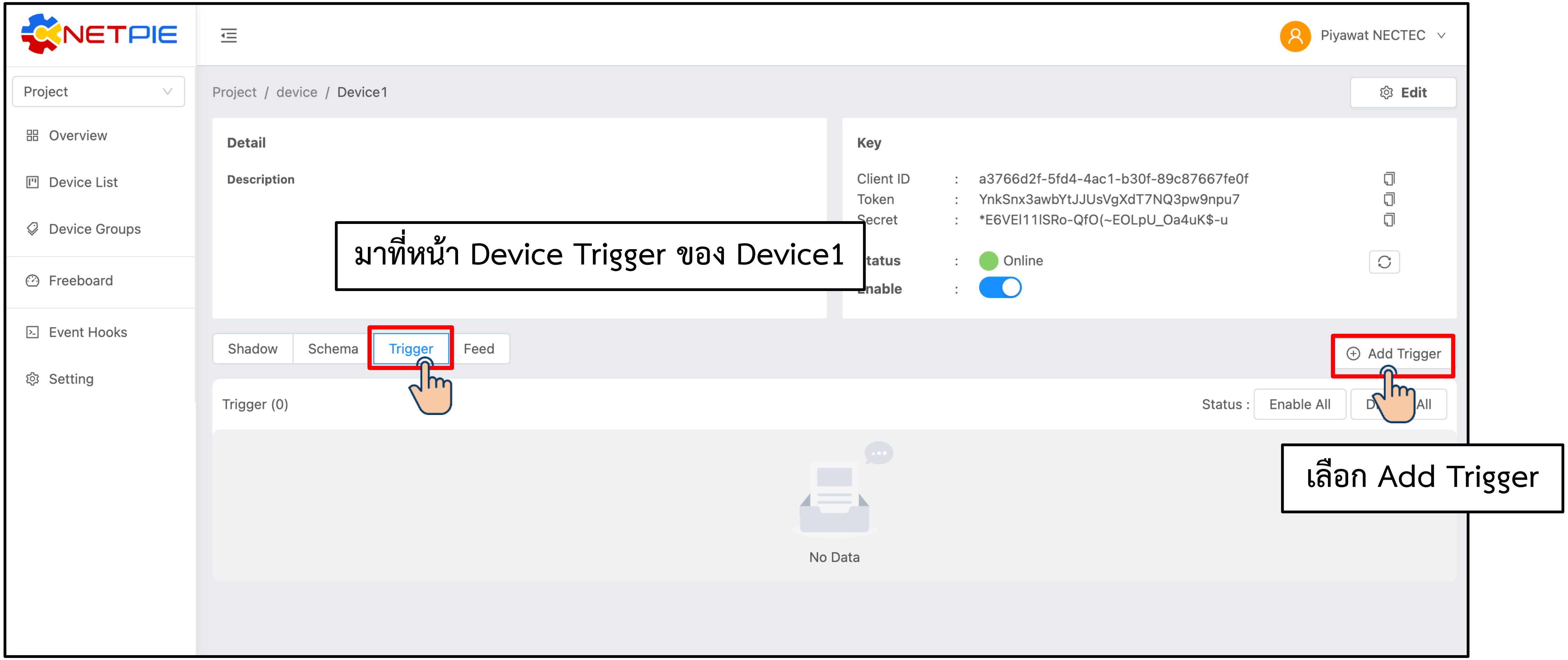
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



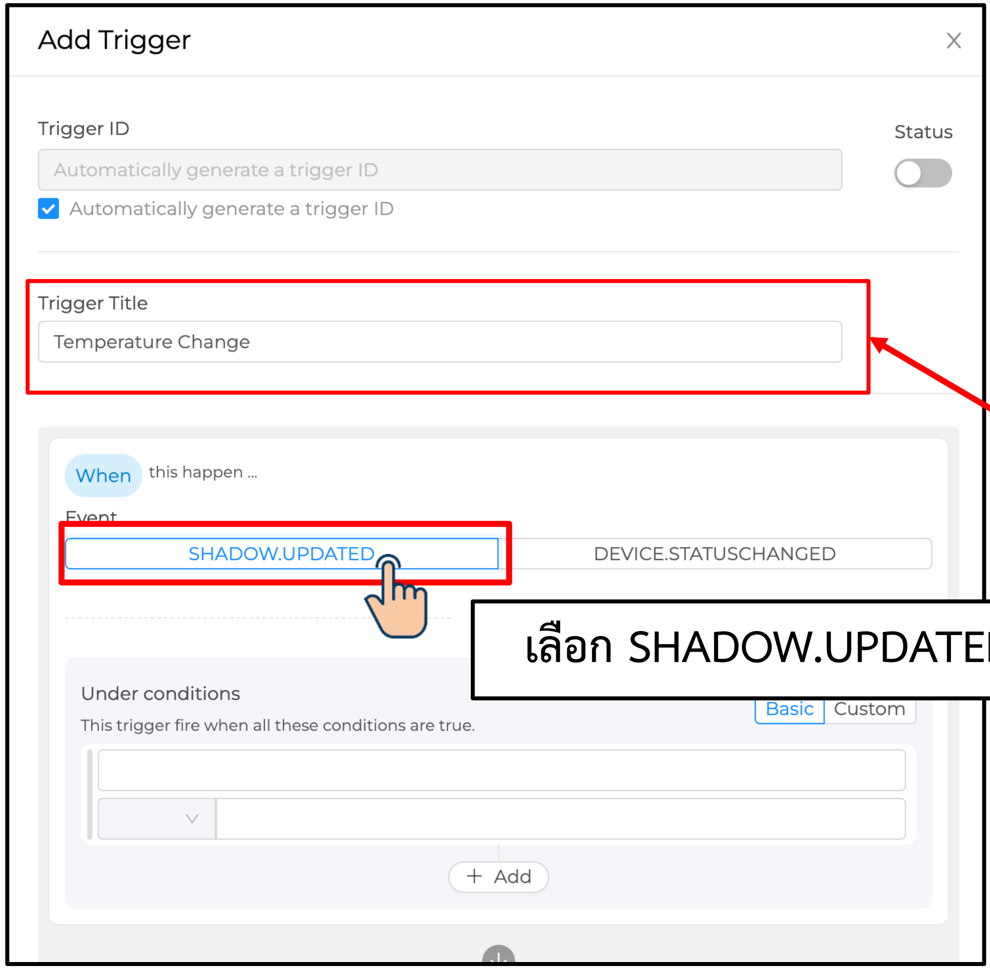
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



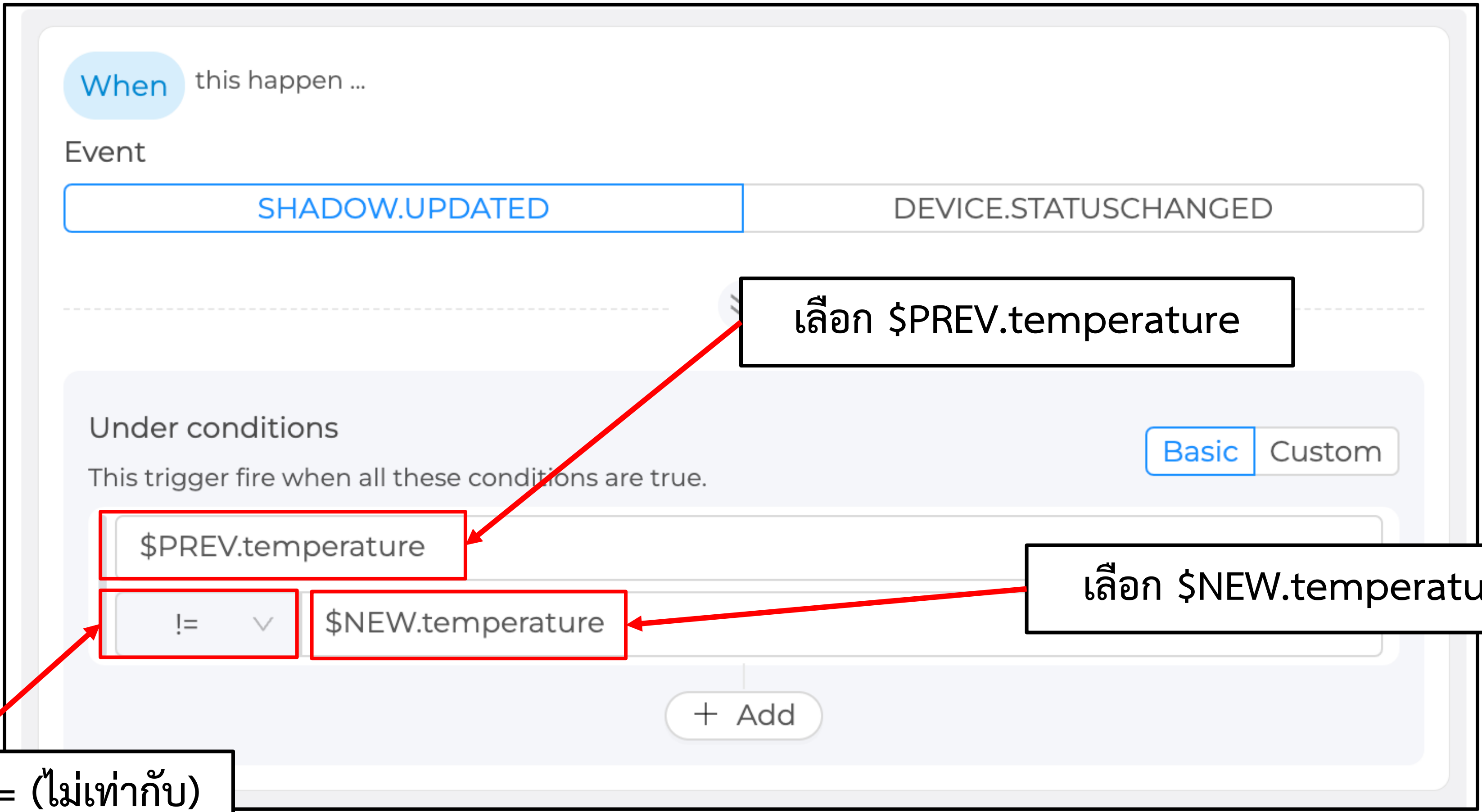
เราจะสร้างเงื่อนไขการแจ้งเตือน คือ เมื่ออุณหภูมิมีการเปลี่ยนแปลง ให้ทำการแจ้งเตือนเข้า Line Notify โดยมีข้อความคือ "Temperature is change from PREV.Temperature °C to NEW.Temperature °C"

ตั้งชื่อ Trigger

เลือก SHADOW.UPDATED

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



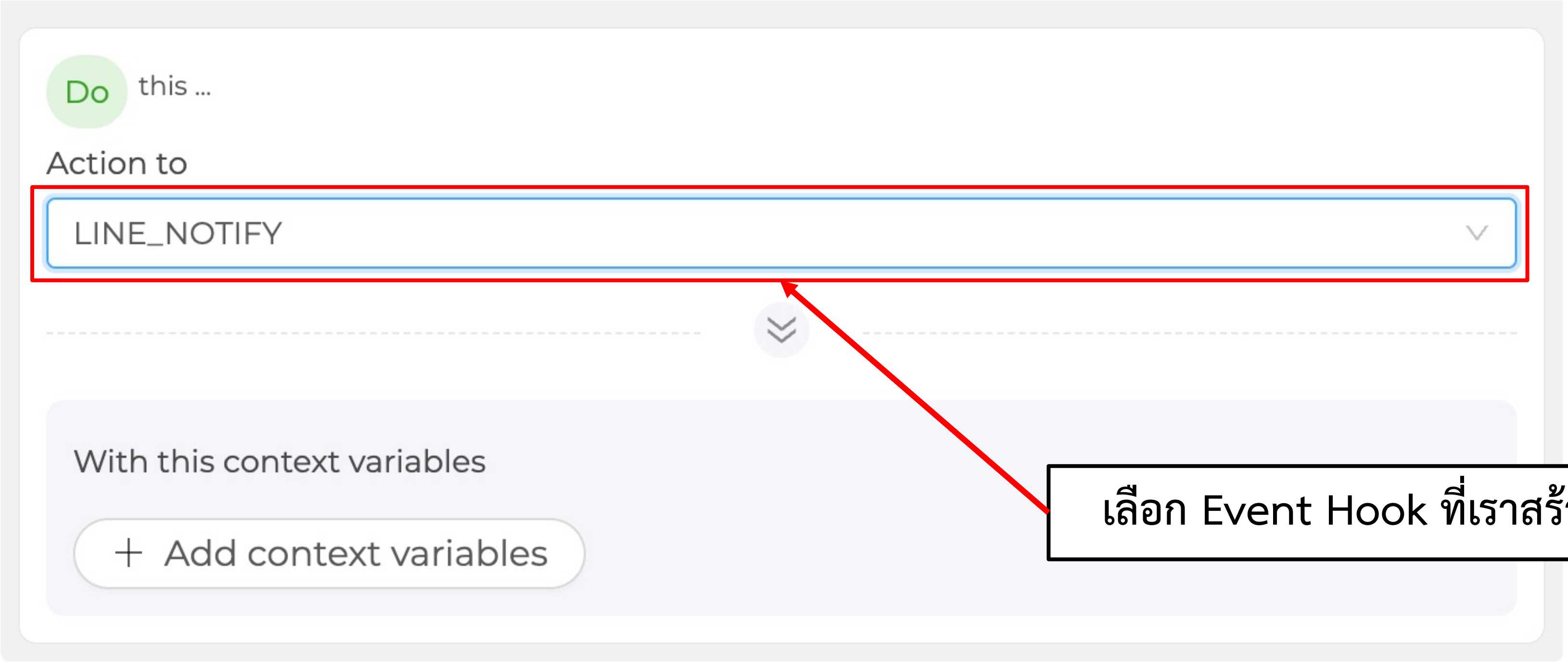
เลือก $PREV.temperature

เลือก $NEW.temperature

เลือกเป็นเงื่อนไข != (ไม่เท่ากับ)

การสร้างการแจ้งเตือนผ่าน Line Notify

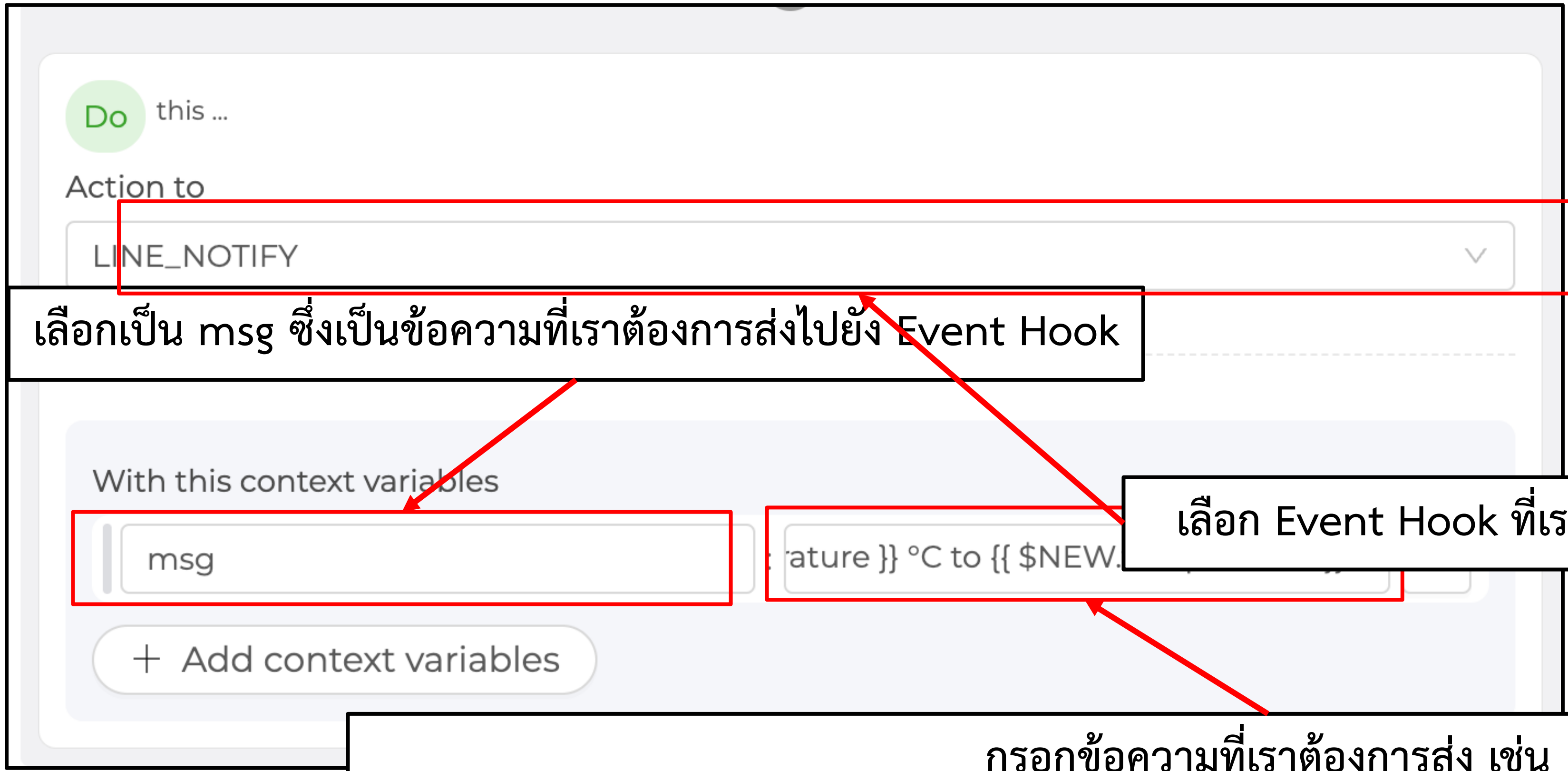
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เลือก Event Hook ที่เราสร้างขึ้นมาในการทดลองที่ 1

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เลือกเป็น msg ซึ่งเป็นข้อความที่เราต้องการส่งไปยัง Event Hook

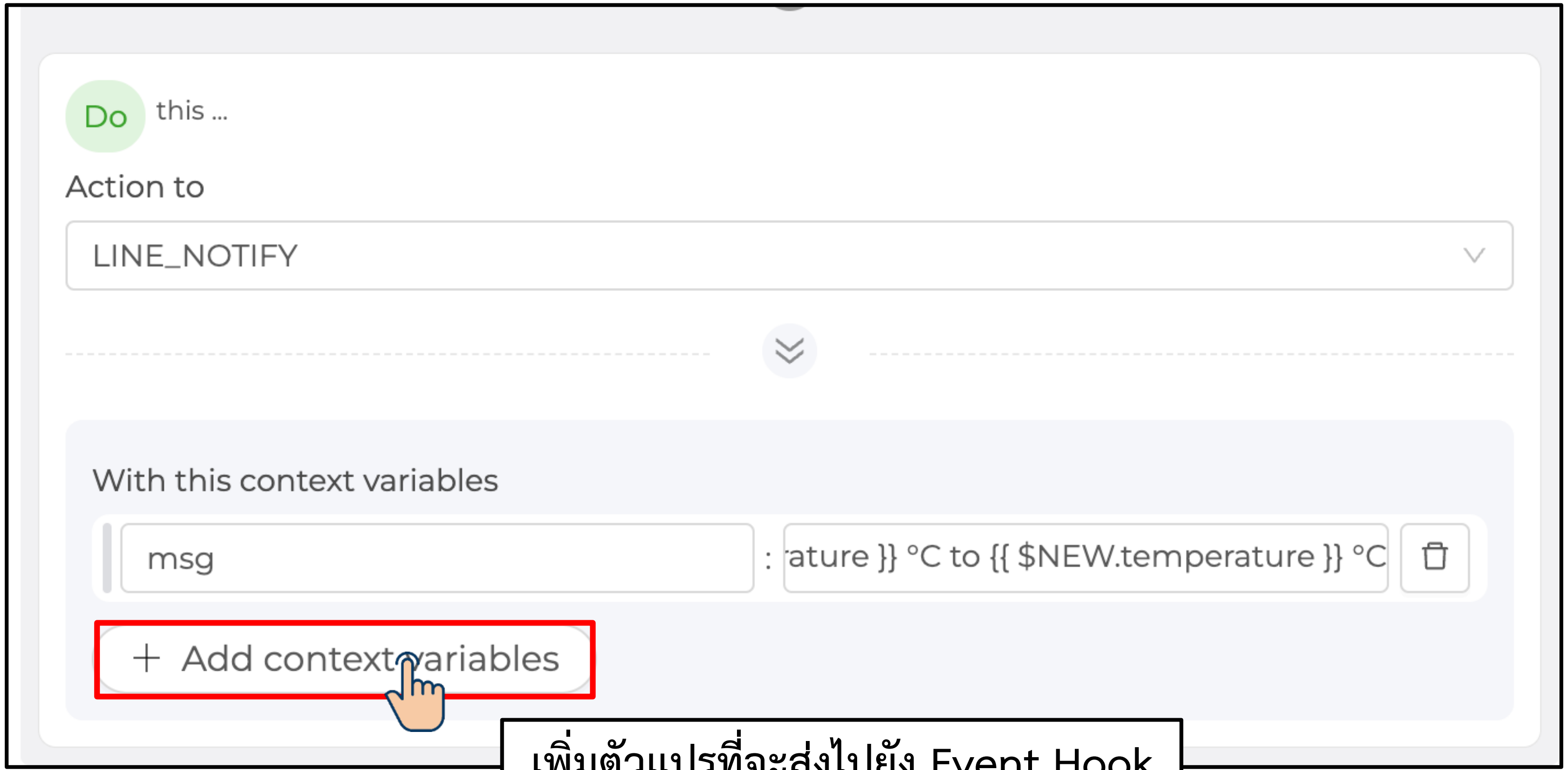
เลือก Event Hook ที่เราสร้างขึ้นมาในการทดลองที่ 1

กรอกข้อความที่เราต้องการส่ง เช่น

Temperature is change from {{ $PREV.temperature }} °C to {{ $NEW.temperature }} °C

การสร้างการแจ้งเตือนผ่าน Line Notify

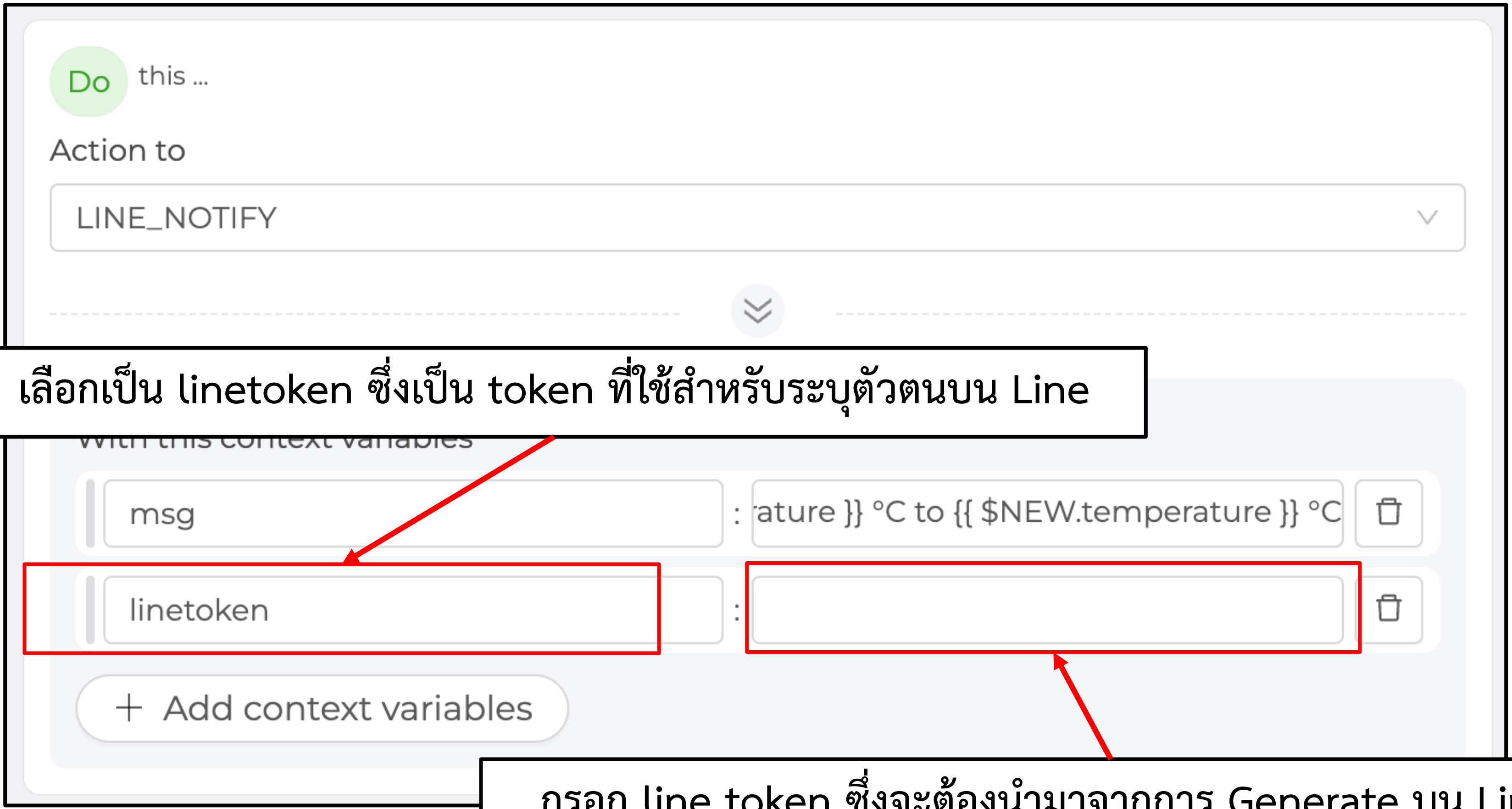
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เพิ่มตัวแปรที่จะส่งไปยัง Event Hook

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เลือกเป็น linetoken ซึ่งเป็น token ที่ใช้สำหรับระบุตัวตนบน Line

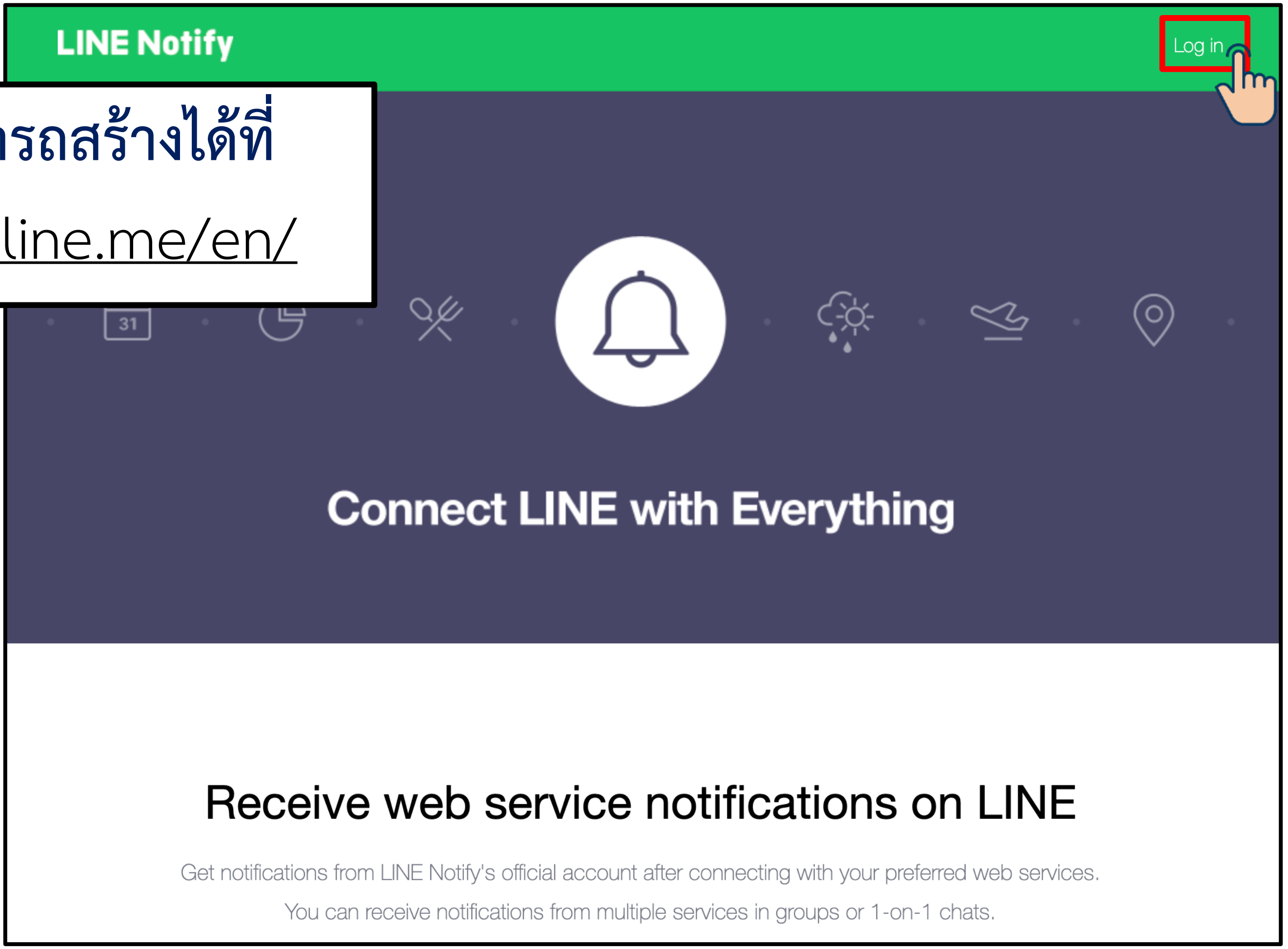
กรอก line token ซึ่งจะต้องนำมาจากการ Generate บน Line Notify

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

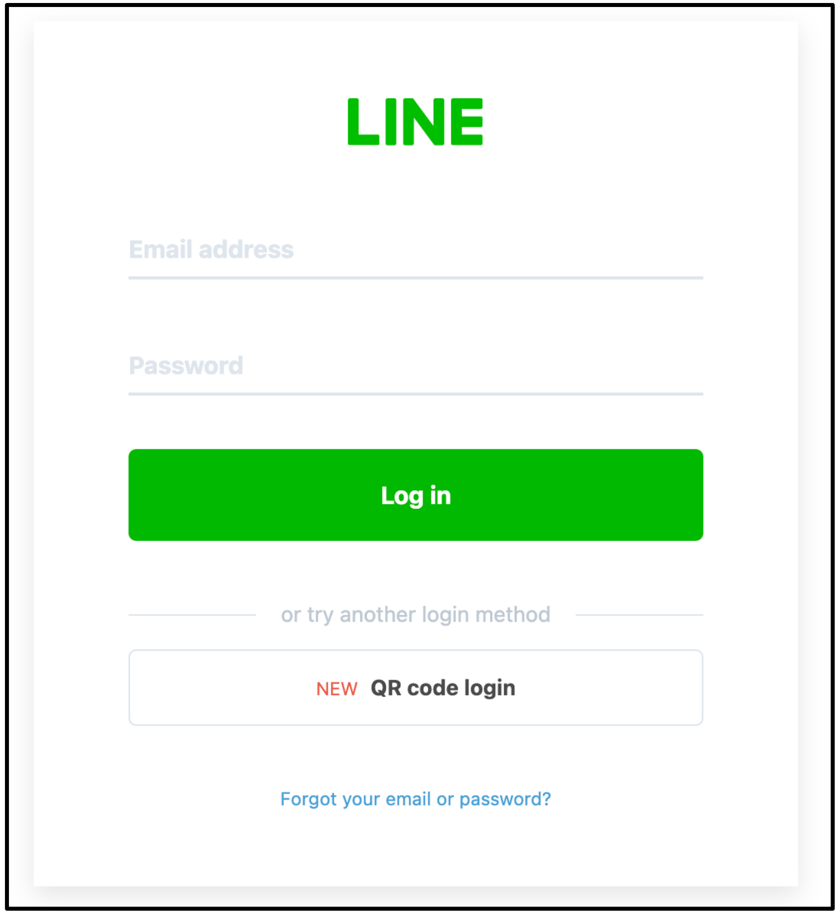
Token Line สามารถสร้างได้ที่

https://notify-bot.line.me/en/



การสร้างการแจ้งเตือนผ่าน Line Notify

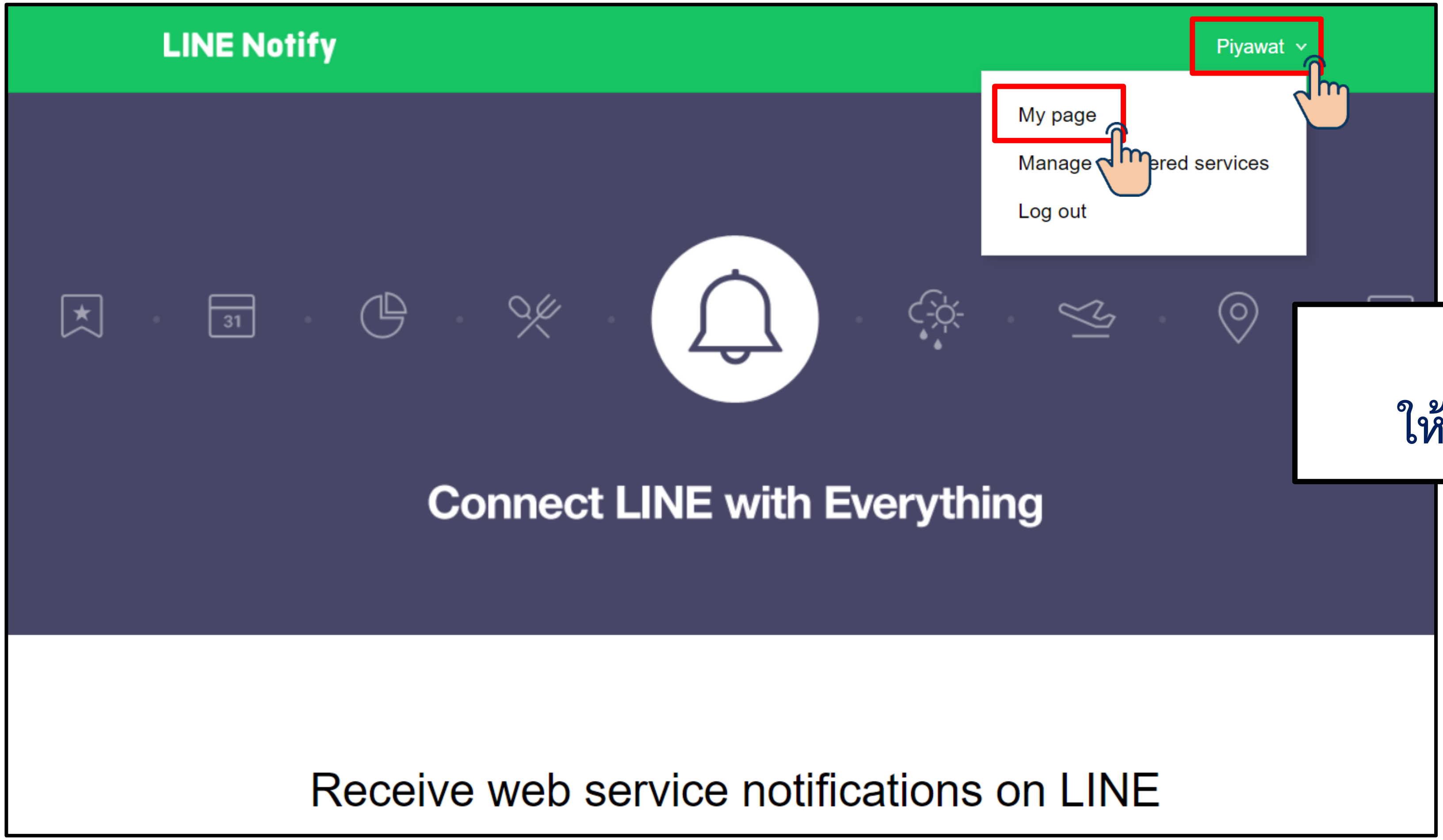
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



Login เข้าสู่ระบบโดยใช้
E-mail และ Password ของ LINE ที่ใช้งาน

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



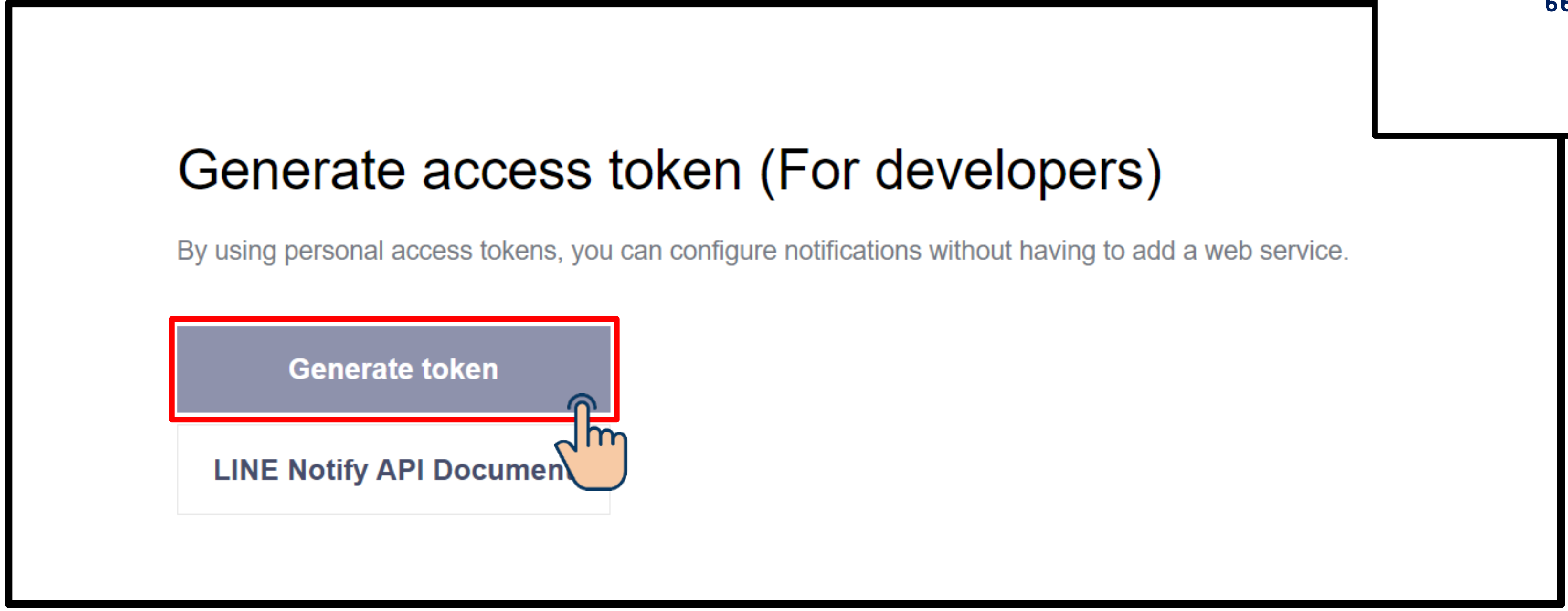
เมื่อเข้าสู่ระบบได้แล้ว
ให้เลือกที่ชื่อ >> My page

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

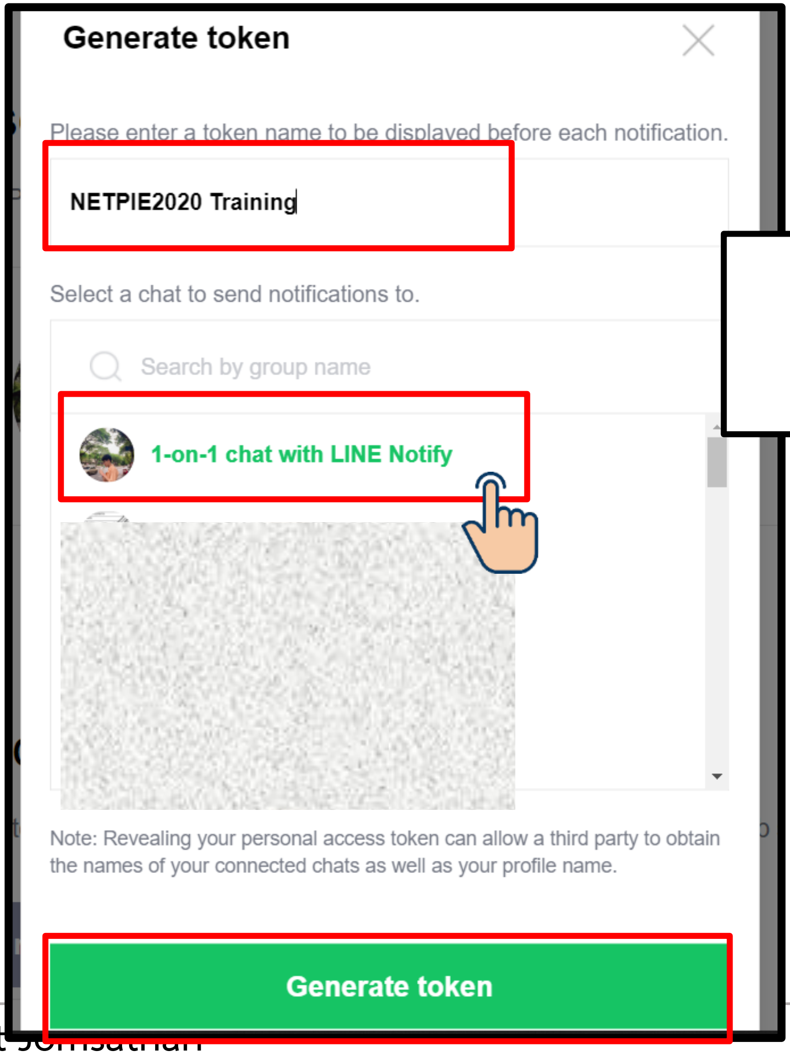
เลื่อนลงมาข้างล่างแล้วเลือก

Generate Token



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

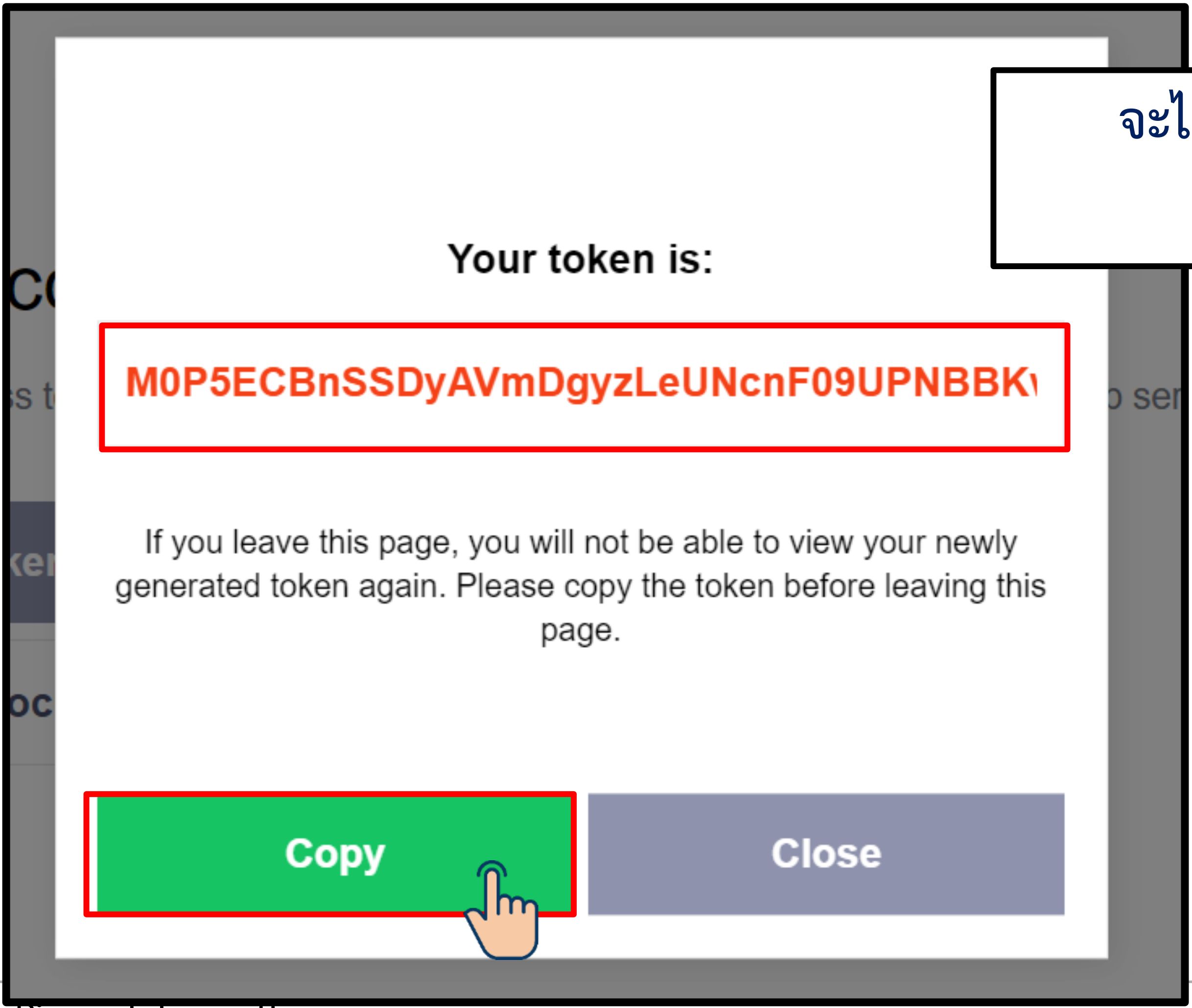


ตั้งชื่อหัวข้อ การสนทนา

แล้วเลือกแชทกับตัวเอง และคลิก Generate Token

การสร้างการแจ้งเตือนผ่าน Line Notify

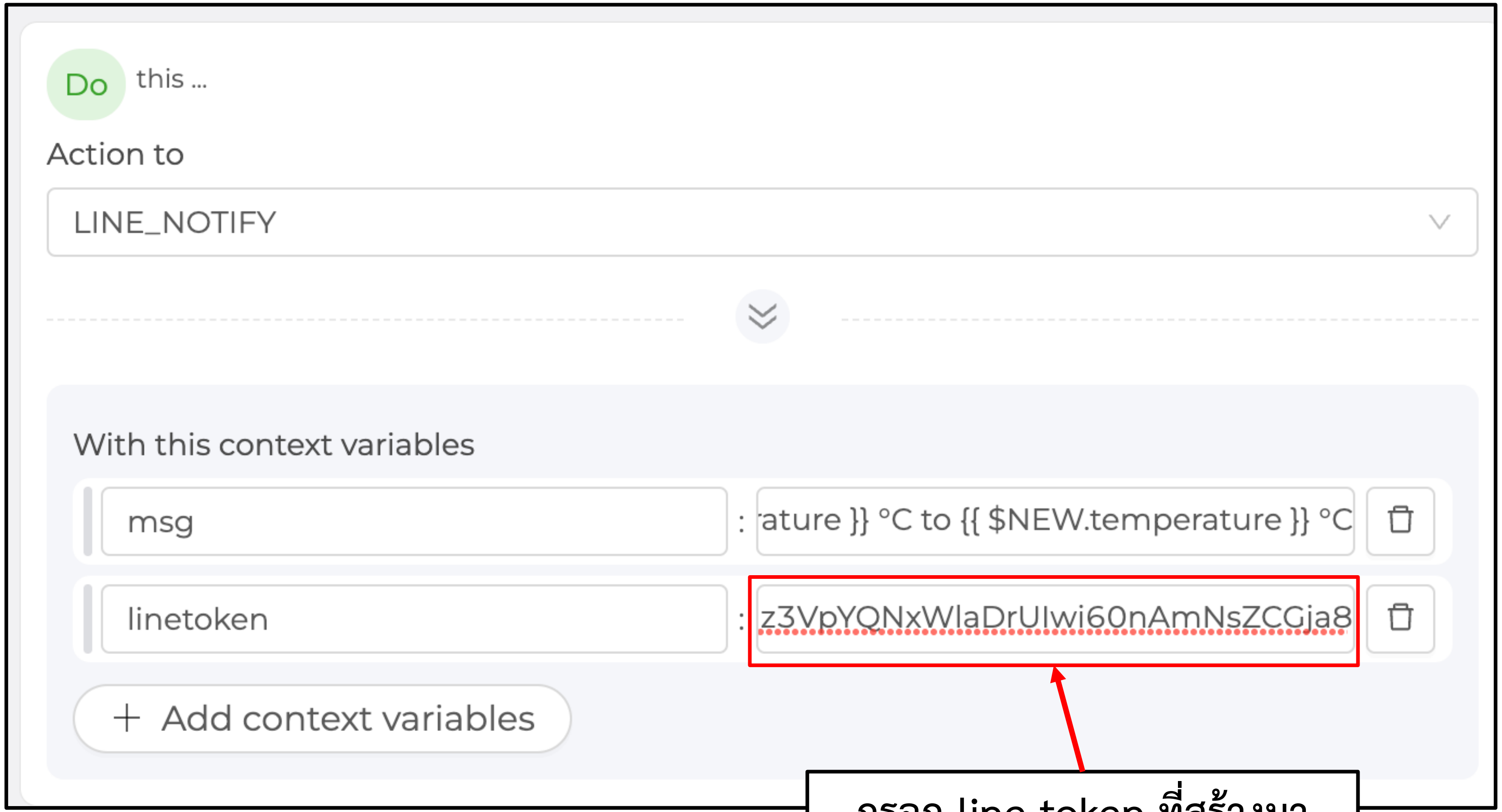
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



จะได้รับ Token ให้ Copy แล้วไปวางใน
linetoken ใน Device Trigger

การสร้างการแจ้งเตือนผ่าน Line Notify

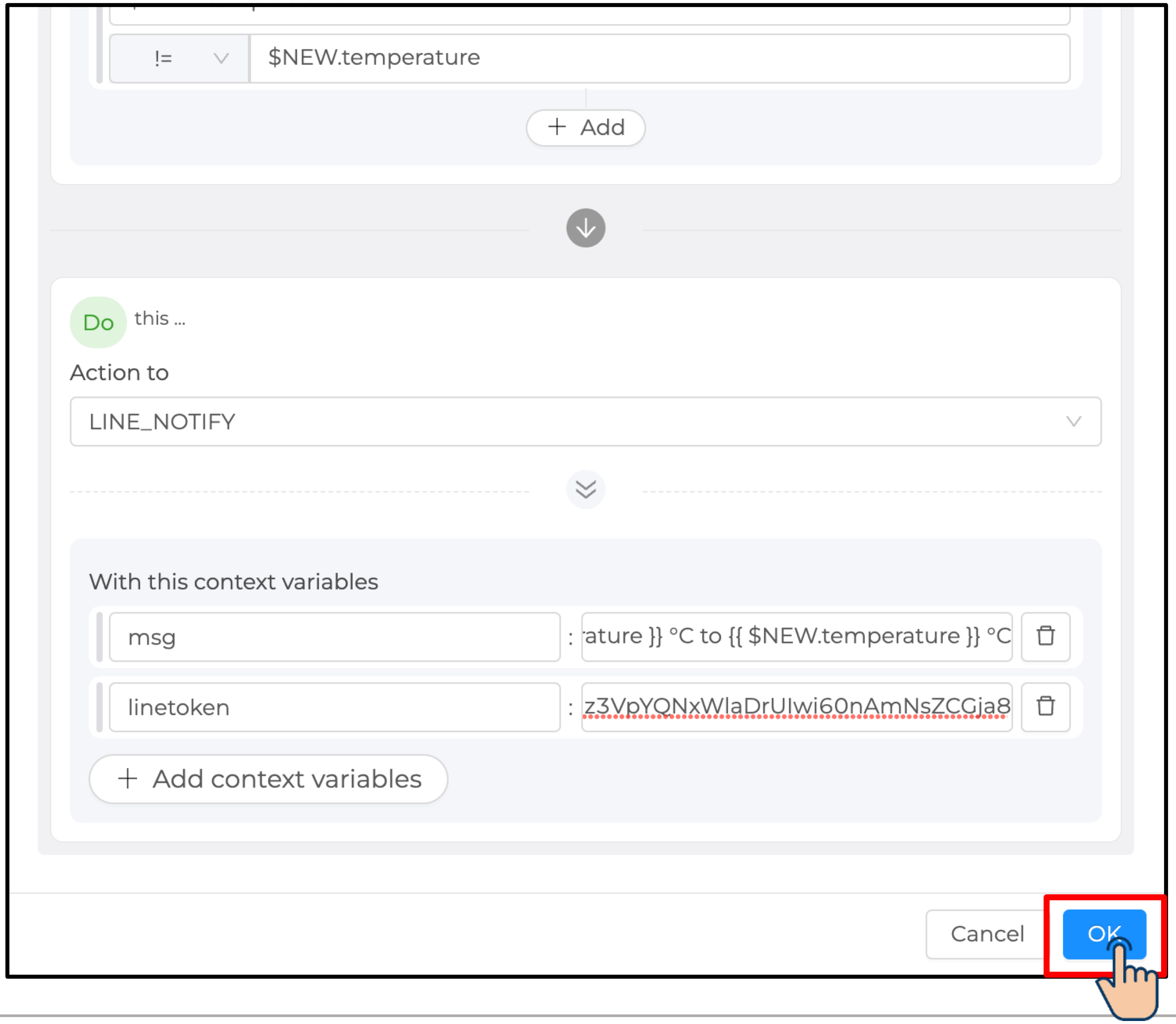
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



กรอก line token ที่สร้างมา

การสร้างการแจ้งเตือนผ่าน Line Notify

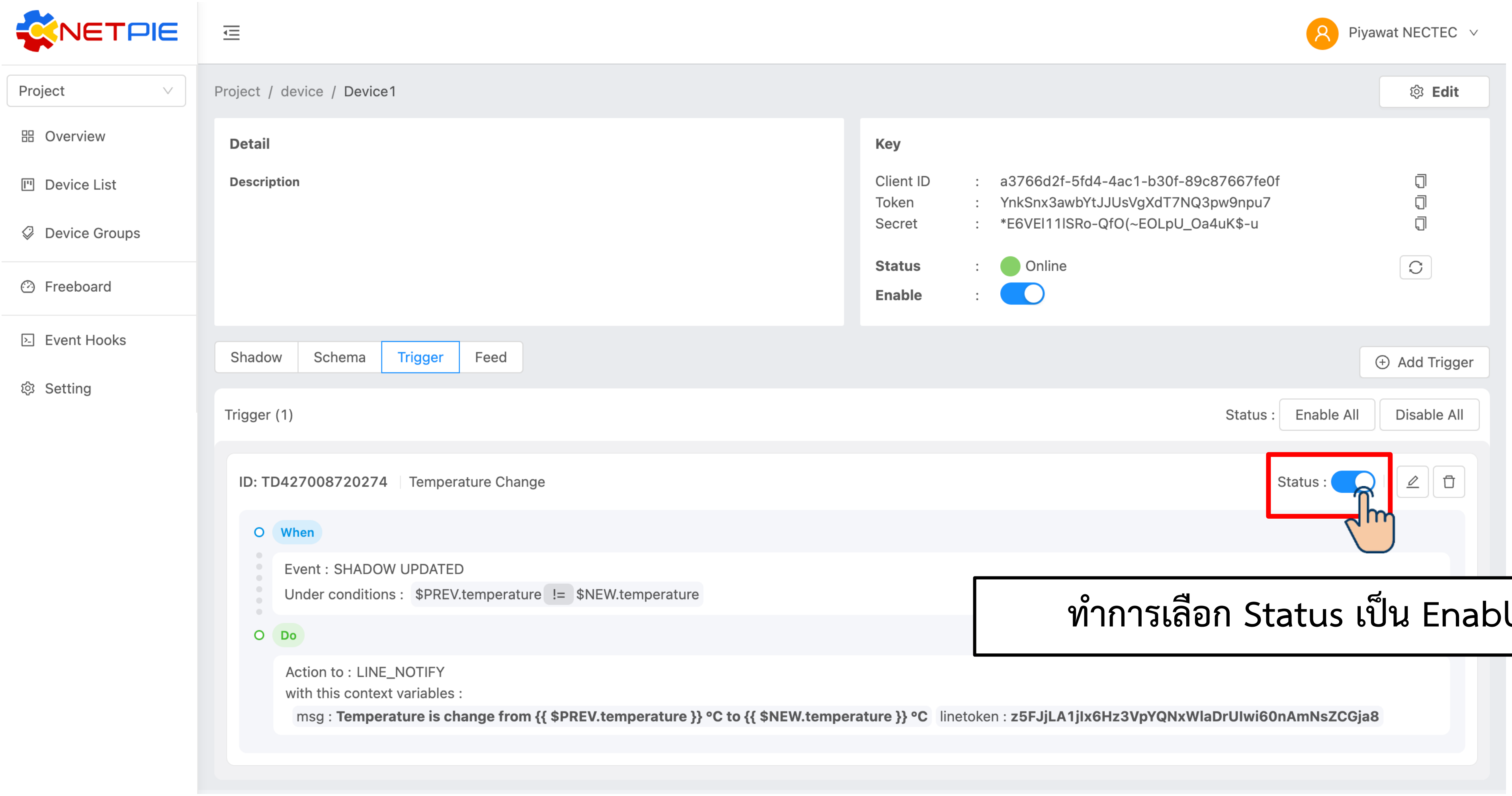
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เมื่อสร้างทั้งหมดเสร็จแล้วทำการเลือก OK

การสร้างการแจ้งเตือนผ่าน Line Notify

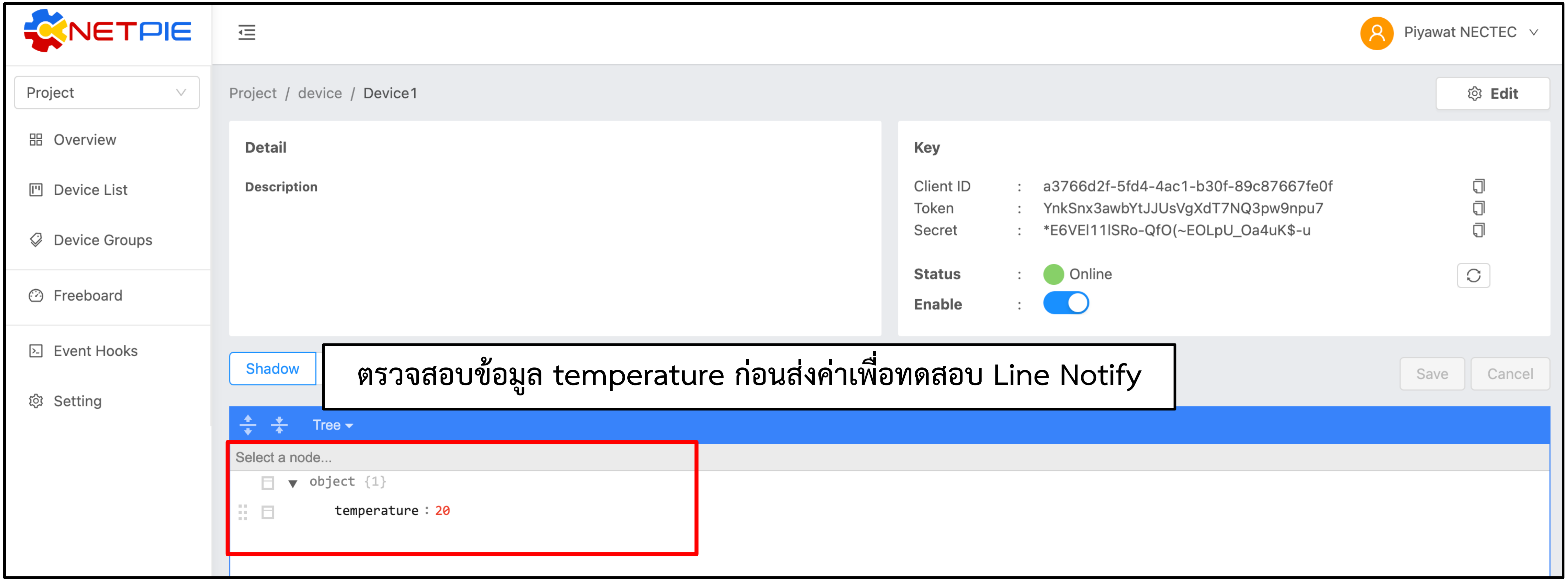
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ทำการเลือก Status เป็น Enable

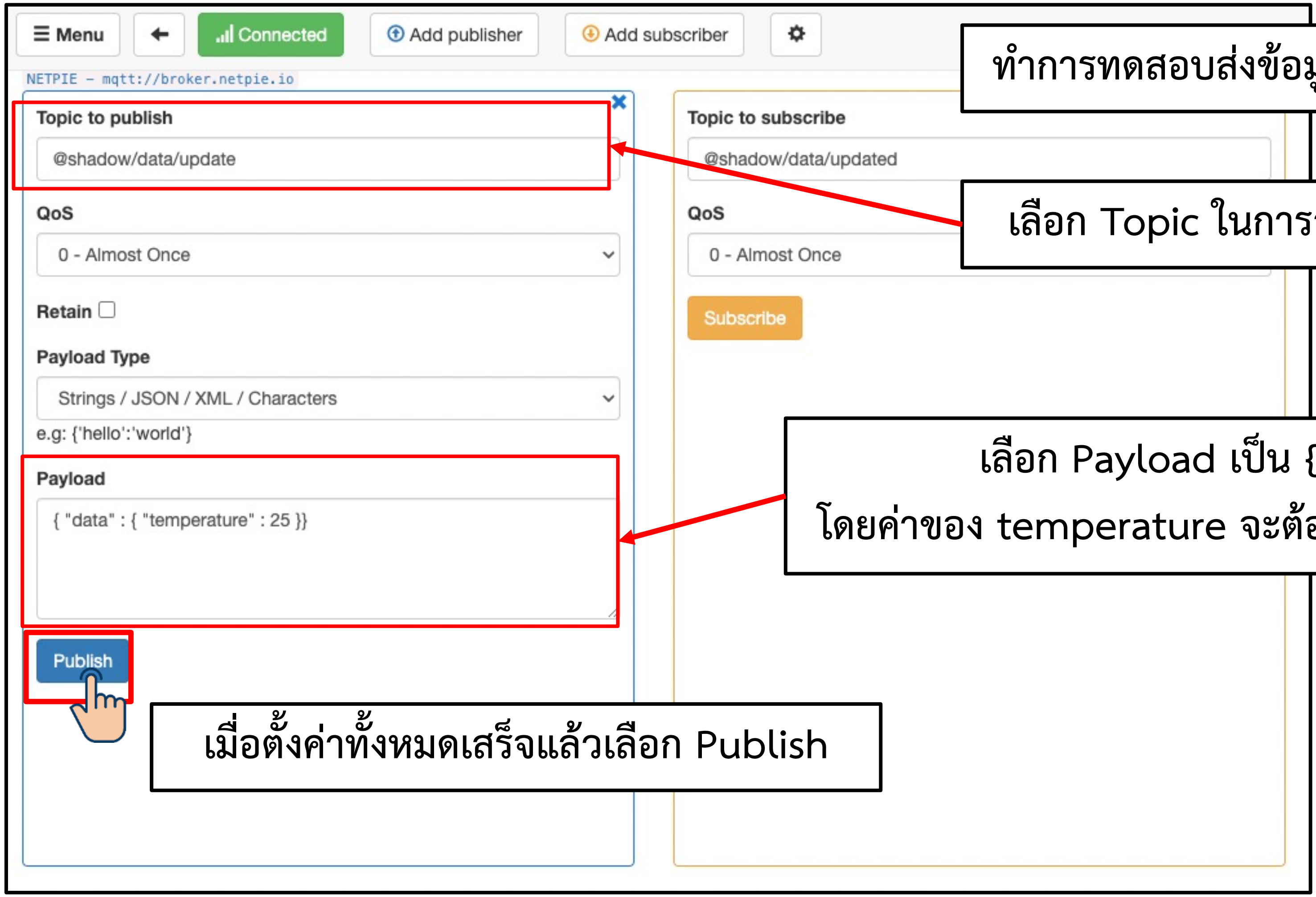
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ทำการทดสอบส่งข้อมูล โดยใช้ MQTT Box

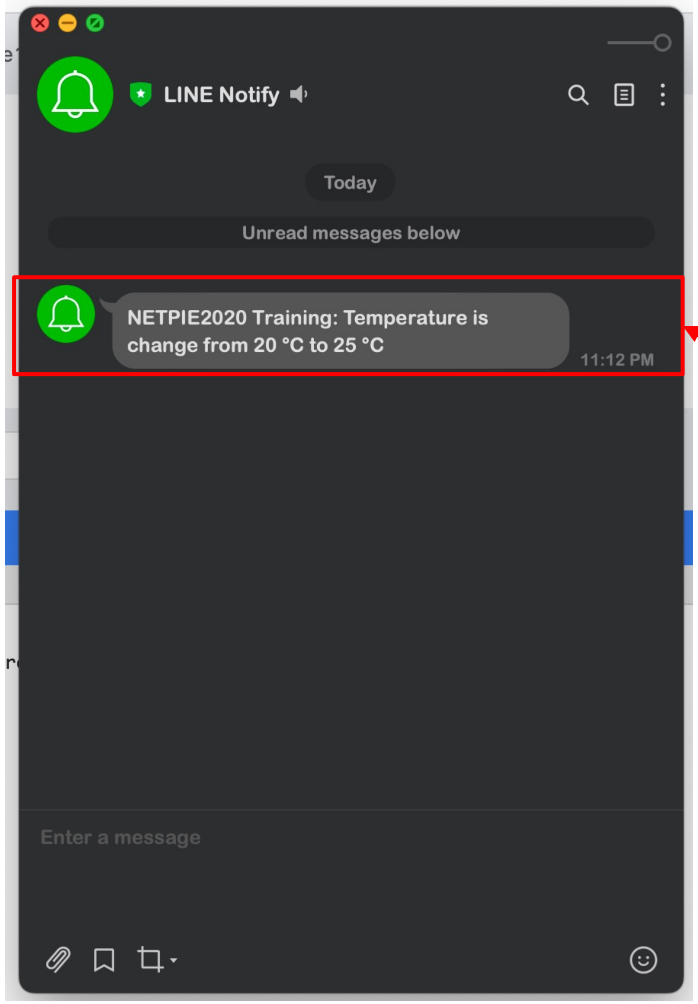
เลือก Topic ในการส่งคือ @shadow/data/update

เลือก Payload เป็น { "data" : { "temperature" : 25 }}
โดยค่าของ temperature จะต้องไม่เท่ากับใน shadow เพื่อให้เกิดการแจ้งเตือน

เมื่อตั้งค่าทั้งหมดเสร็จแล้วเลือก Publish

การสร้างการแจ้งเตือนผ่าน Line Notify

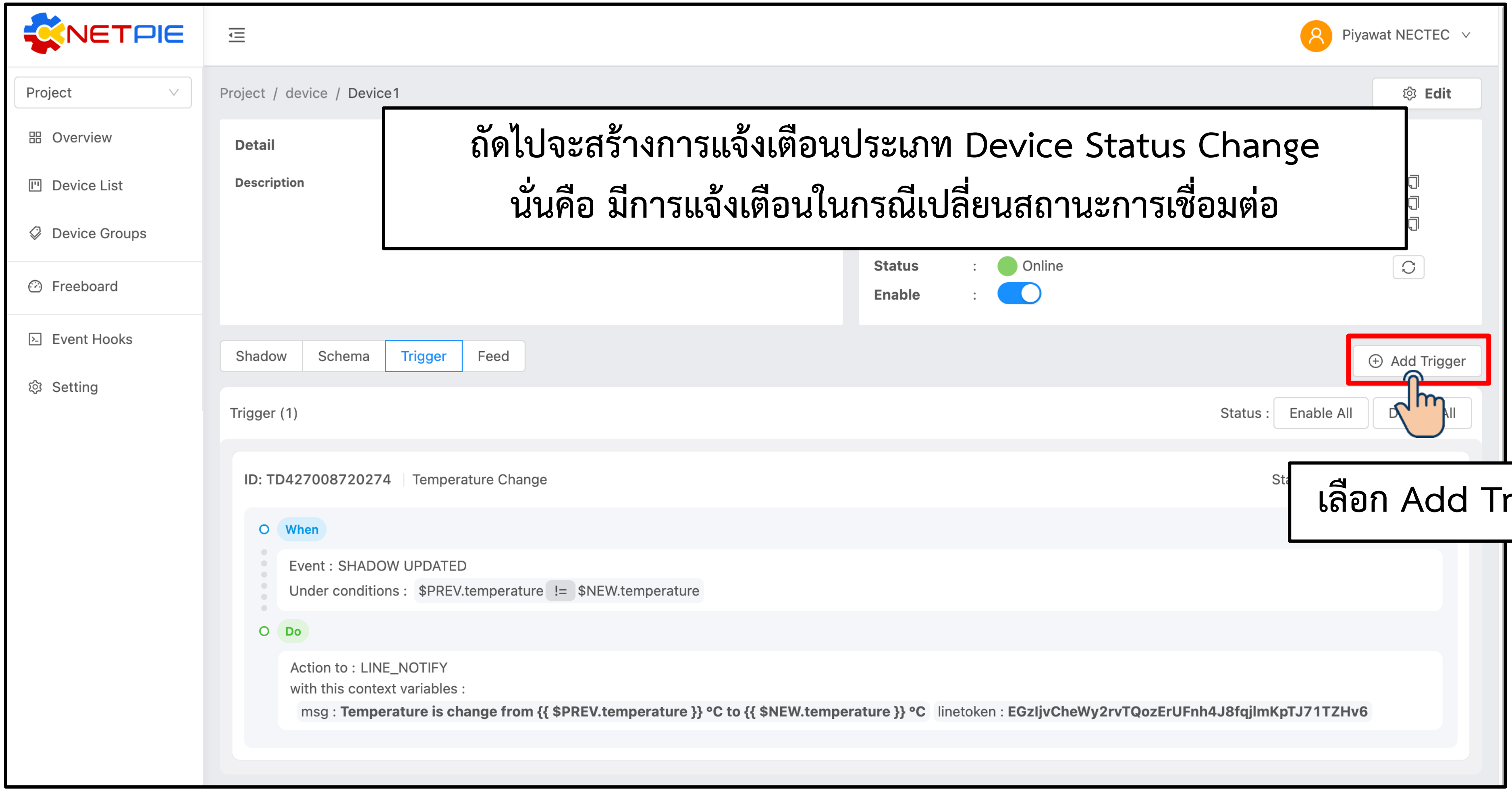
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ข้อความที่ถูกส่งเข้ามาใน Line Notify

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

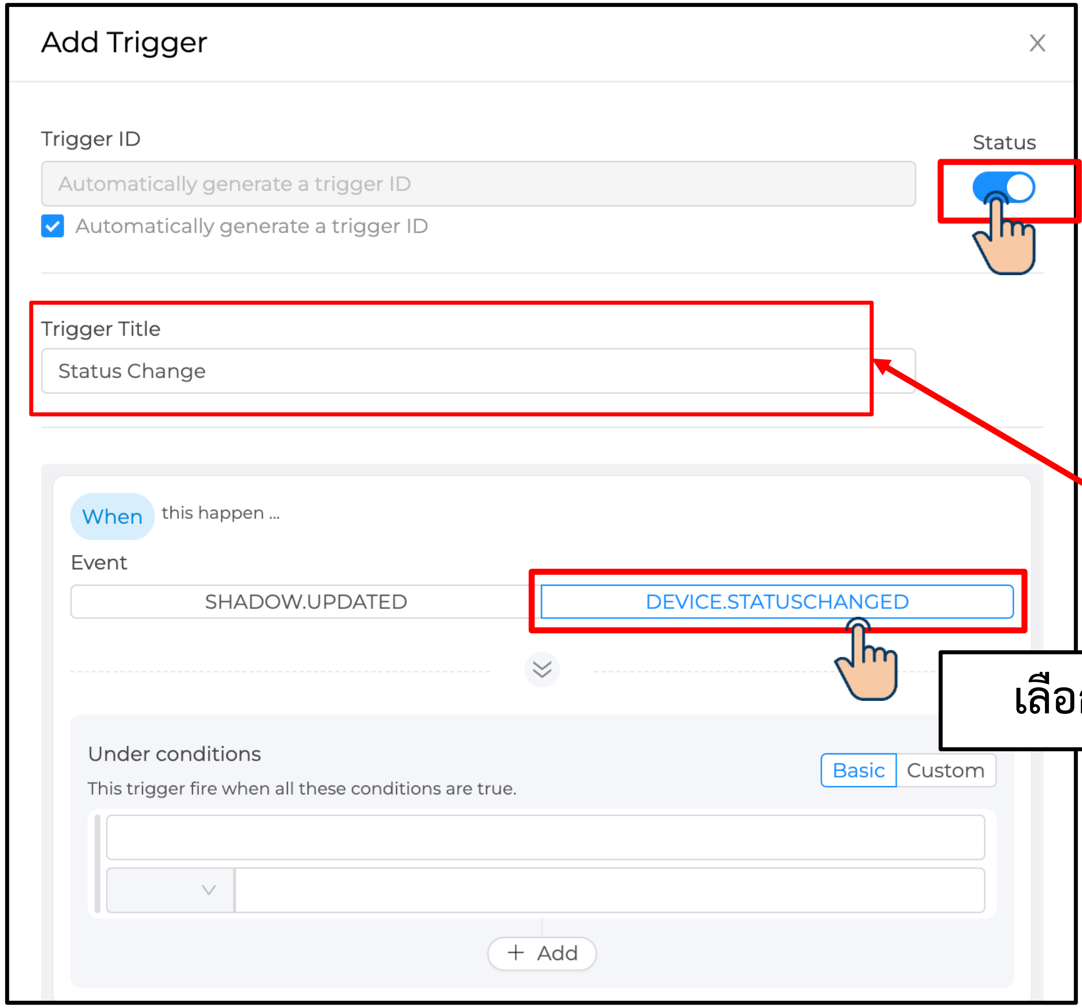


ถัดไปจะสร้างการแจ้งเตือนประเภท Device Status Change
นั่นคือ มีการแจ้งเตือนในกรณีเปลี่ยนสถานะการเชื่อมต่อ

เลือก Add Trigger

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



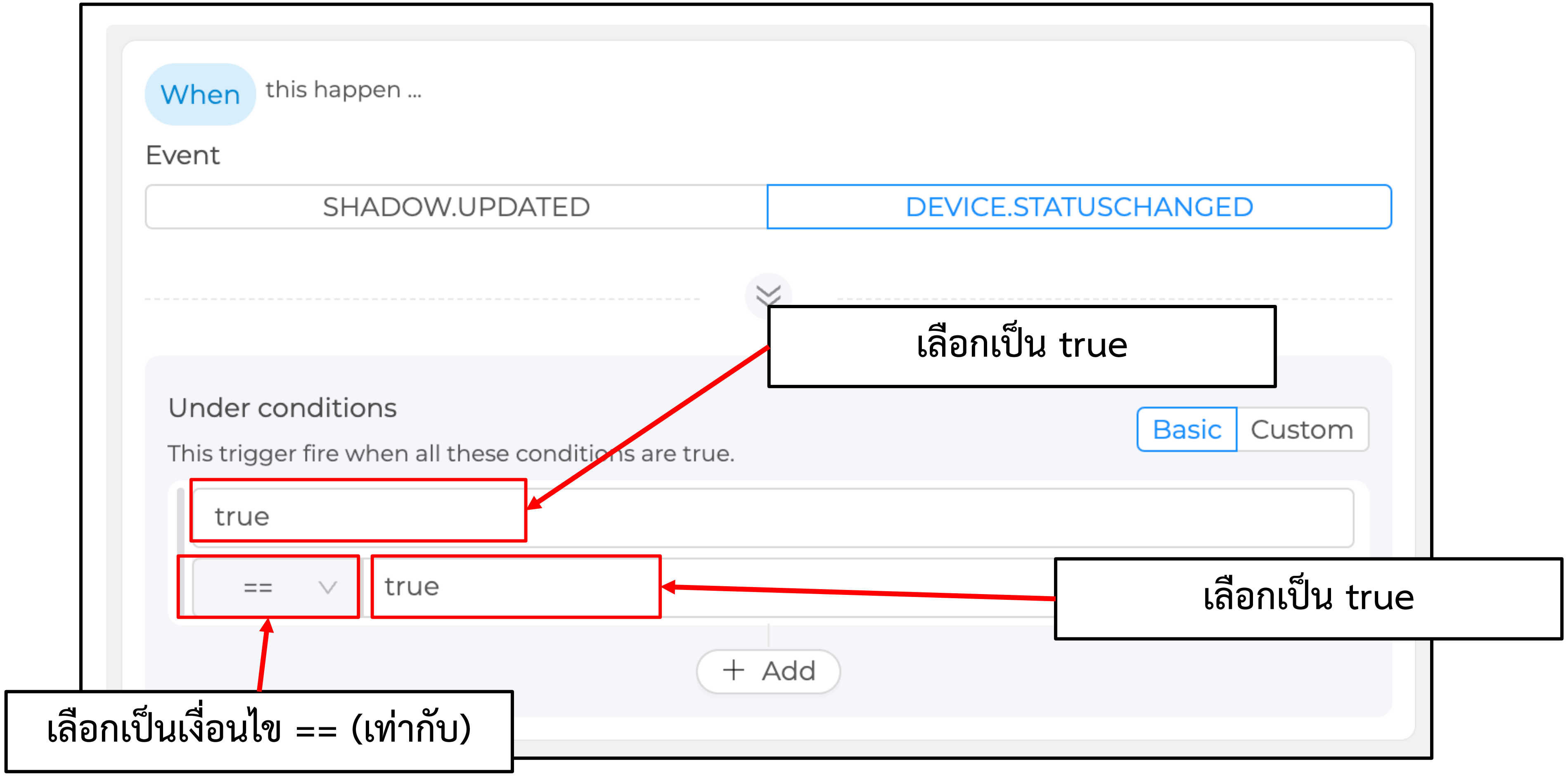
เราจะสร้างเงื่อนไขการแจ้งเตือน คือ เมื่อมีการเปลี่ยนสถานะไม่ว่าจะเป็นรูปแบบไหนก็ตาม ให้แจ้งเตือนไปยัง Line Notify โดยมีข้อความคือ "Status change from OLD.Status to NEW.Status"

ตั้งชื่อ Trigger

เลือก DEVICE.STATUSCHANGED

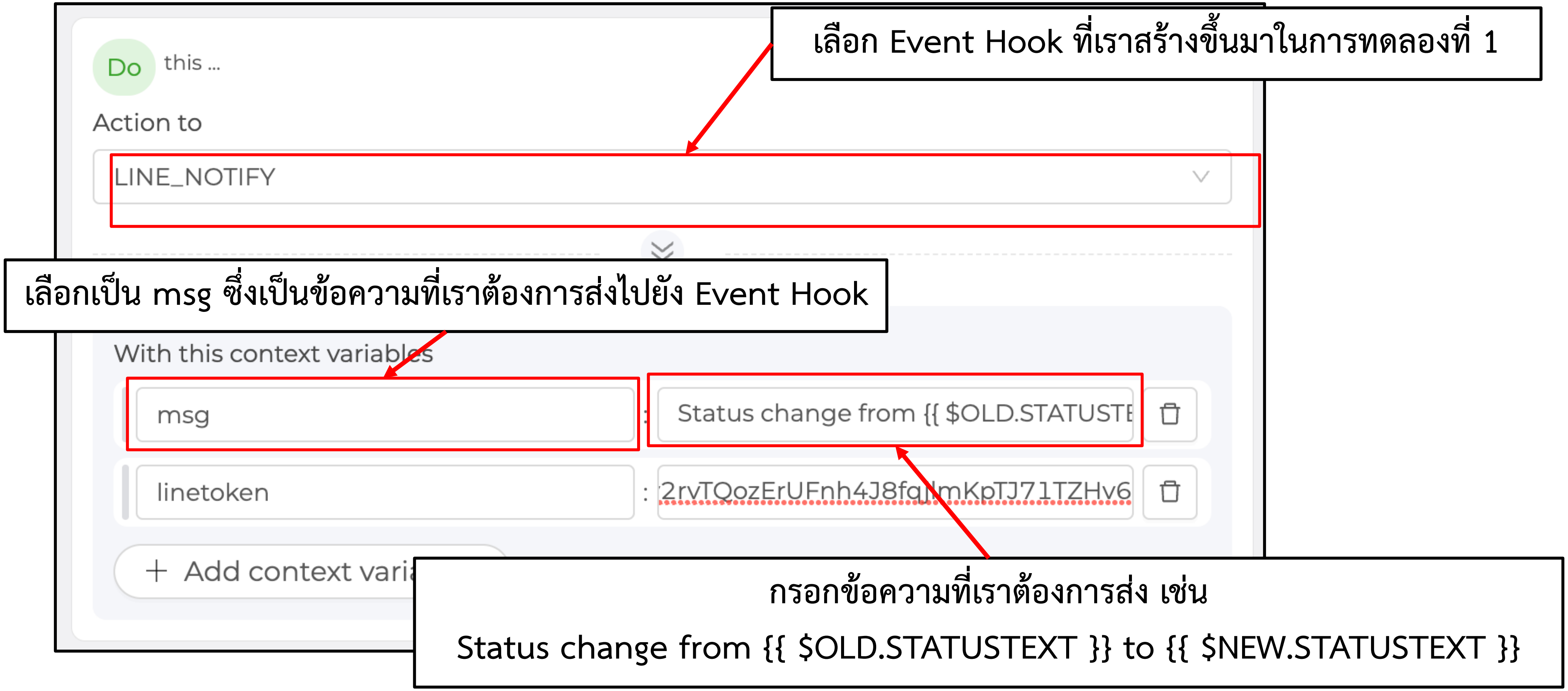
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



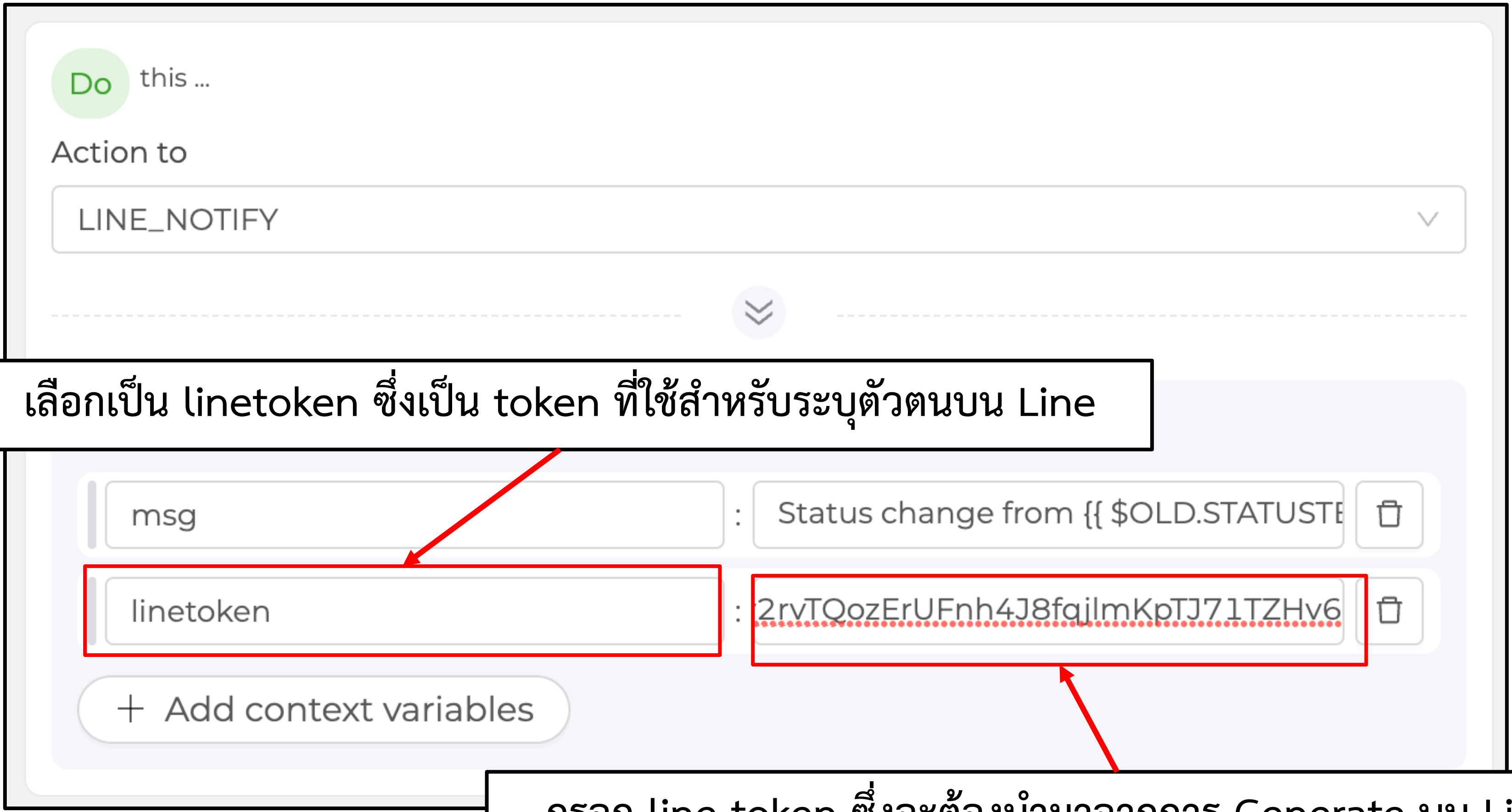
การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

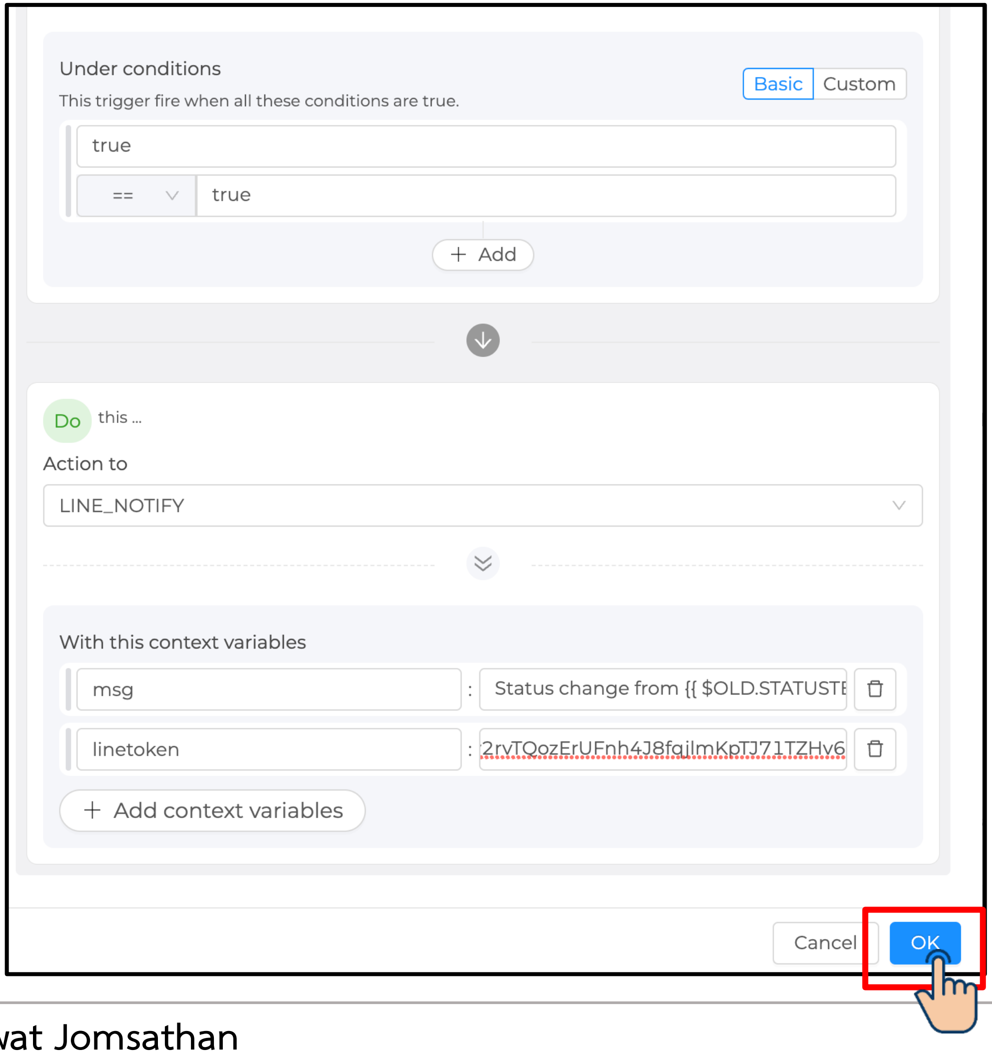


เลือกเป็น linetoken ซึ่งเป็น token ที่ใช้สำหรับระบุตัวตนบน Line

กรอก line token ซึ่งจะต้องนำมาจากการ Generate บน Line Notify

การสร้างการแจ้งเตือนผ่าน Line Notify

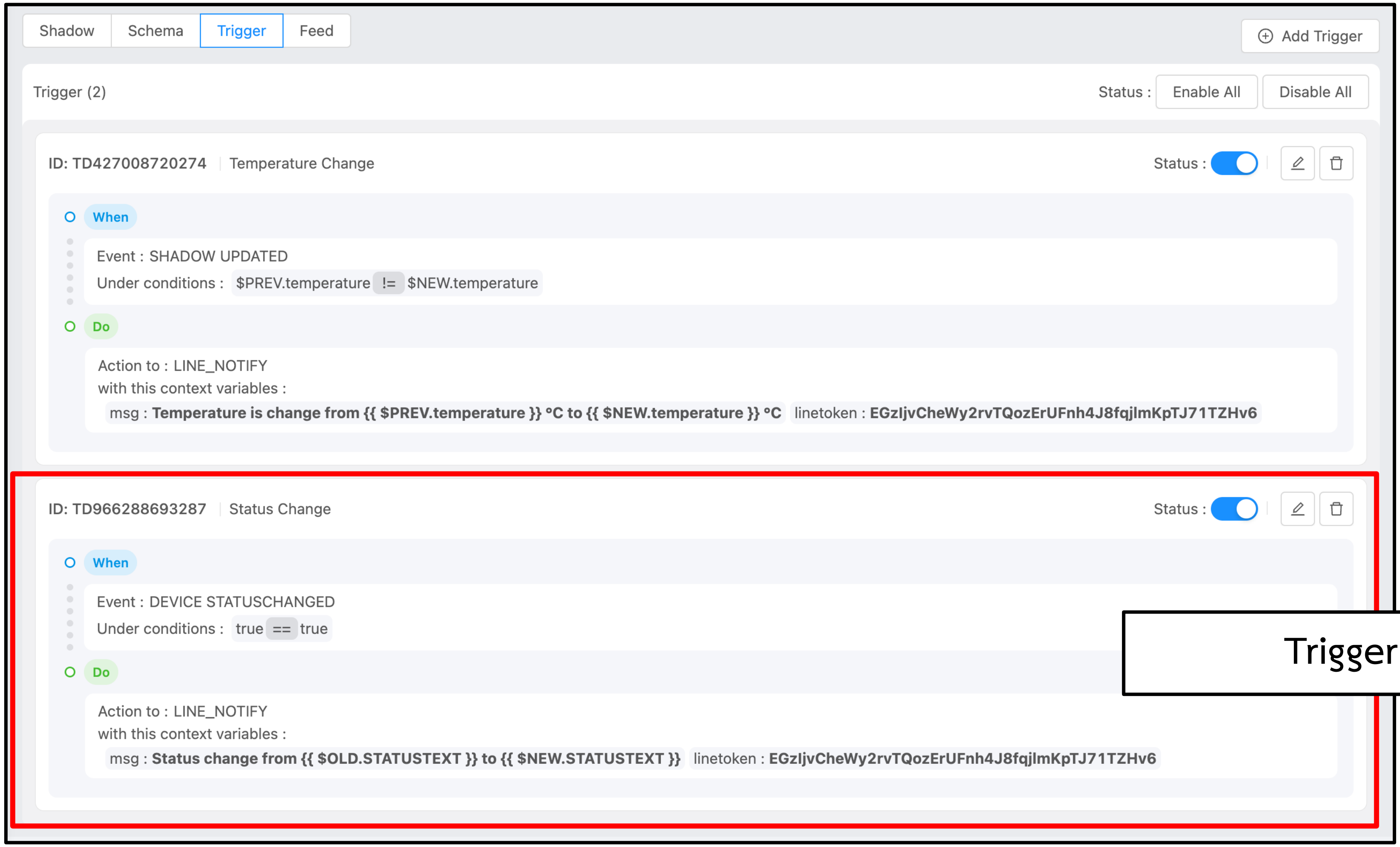
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เมื่อสร้างทั้งหมดเสร็จแล้วทำการเลือก OK

การสร้างการแจ้งเตือนผ่าน Line Notify

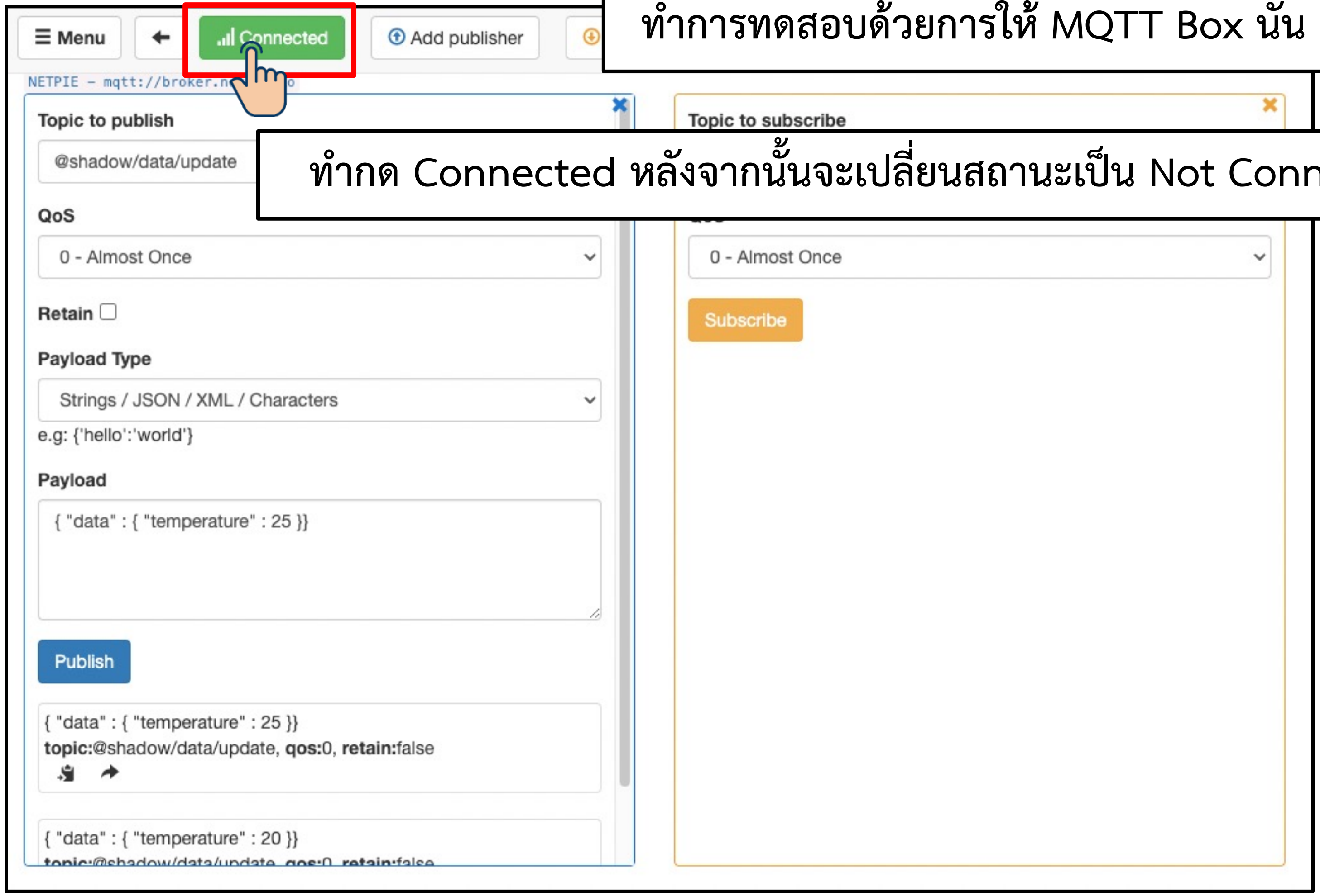
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



Trigger ที่สร้างขึ้นมาใหม่

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify

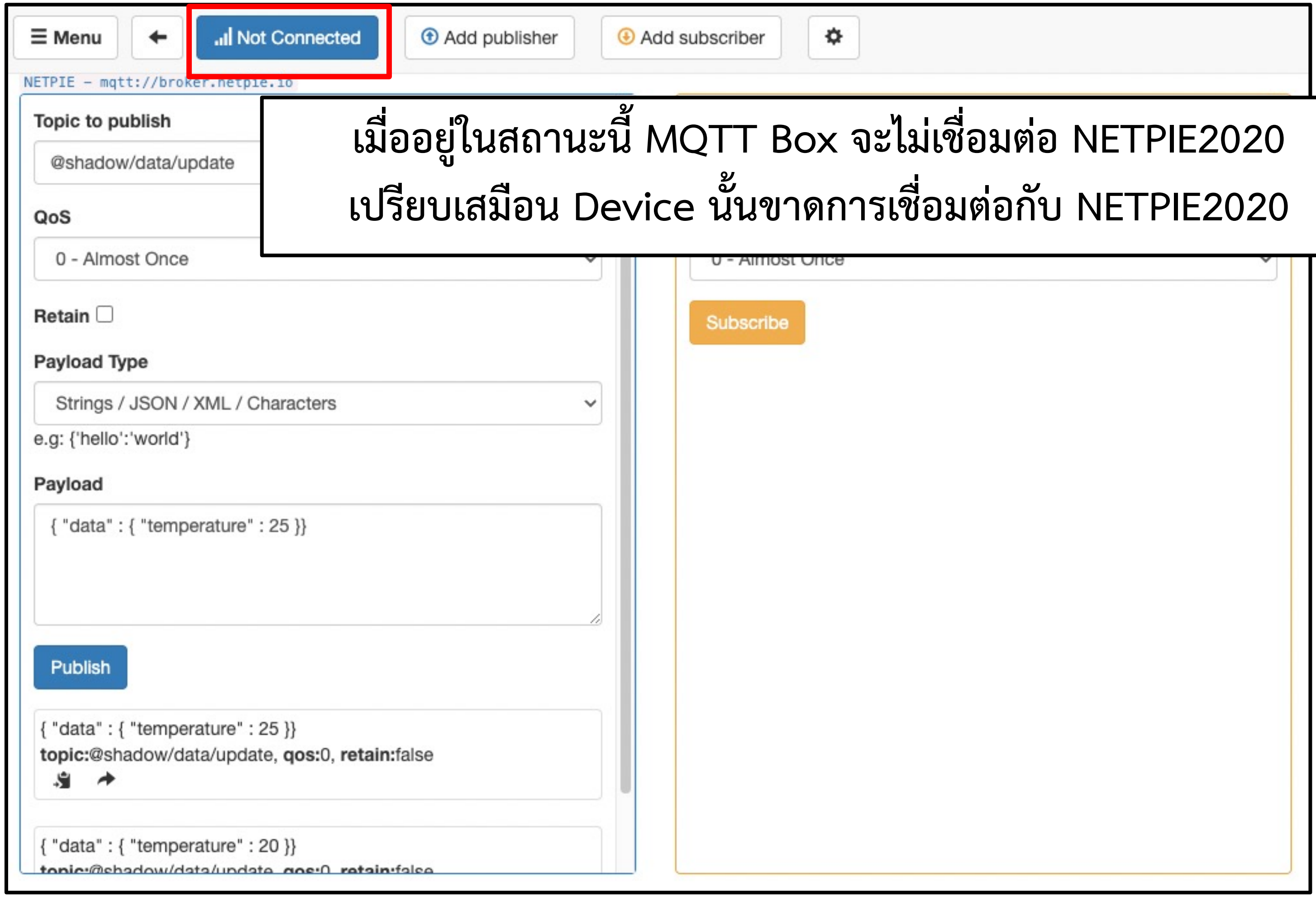


ทำการทดสอบด้วยการให้ MQTT Box นั้น Disconnected

ทำกด Connected หลังจากนั้นจะเปลี่ยนสถานะเป็น Not Connected

การสร้างการแจ้งเตือนผ่าน Line Notify

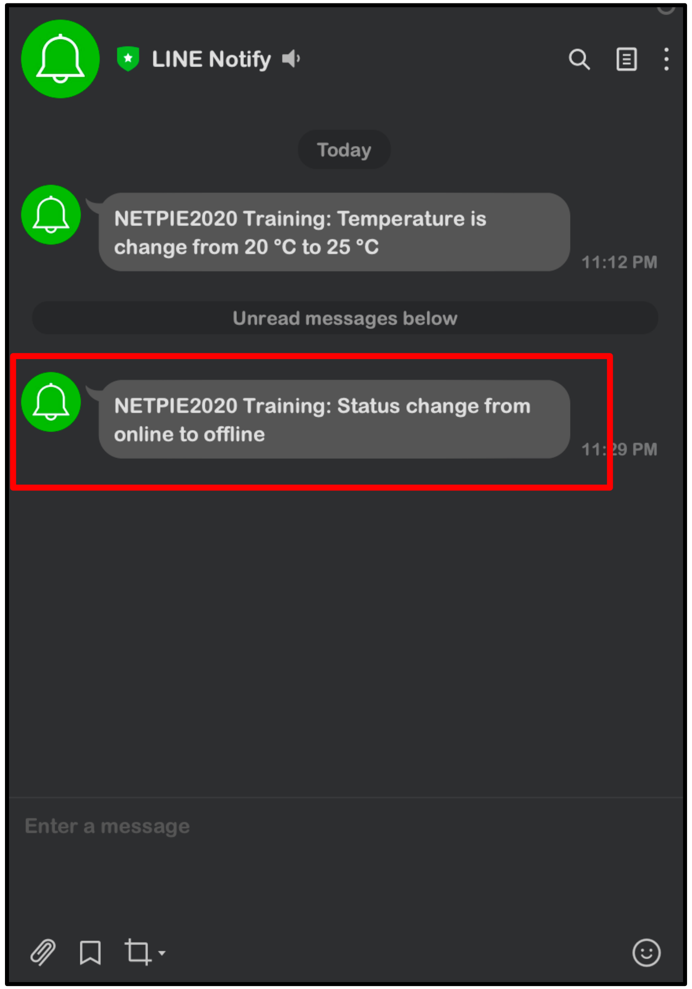
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เมื่ออยู่ในสถานะนี้ MQTT Box จะไม่เชื่อมต่อ NETPIE2020
เปรียบเสมือน Device นั้นขาดการเชื่อมต่อกับ NETPIE2020

การสร้างการแจ้งเตือนผ่าน Line Notify

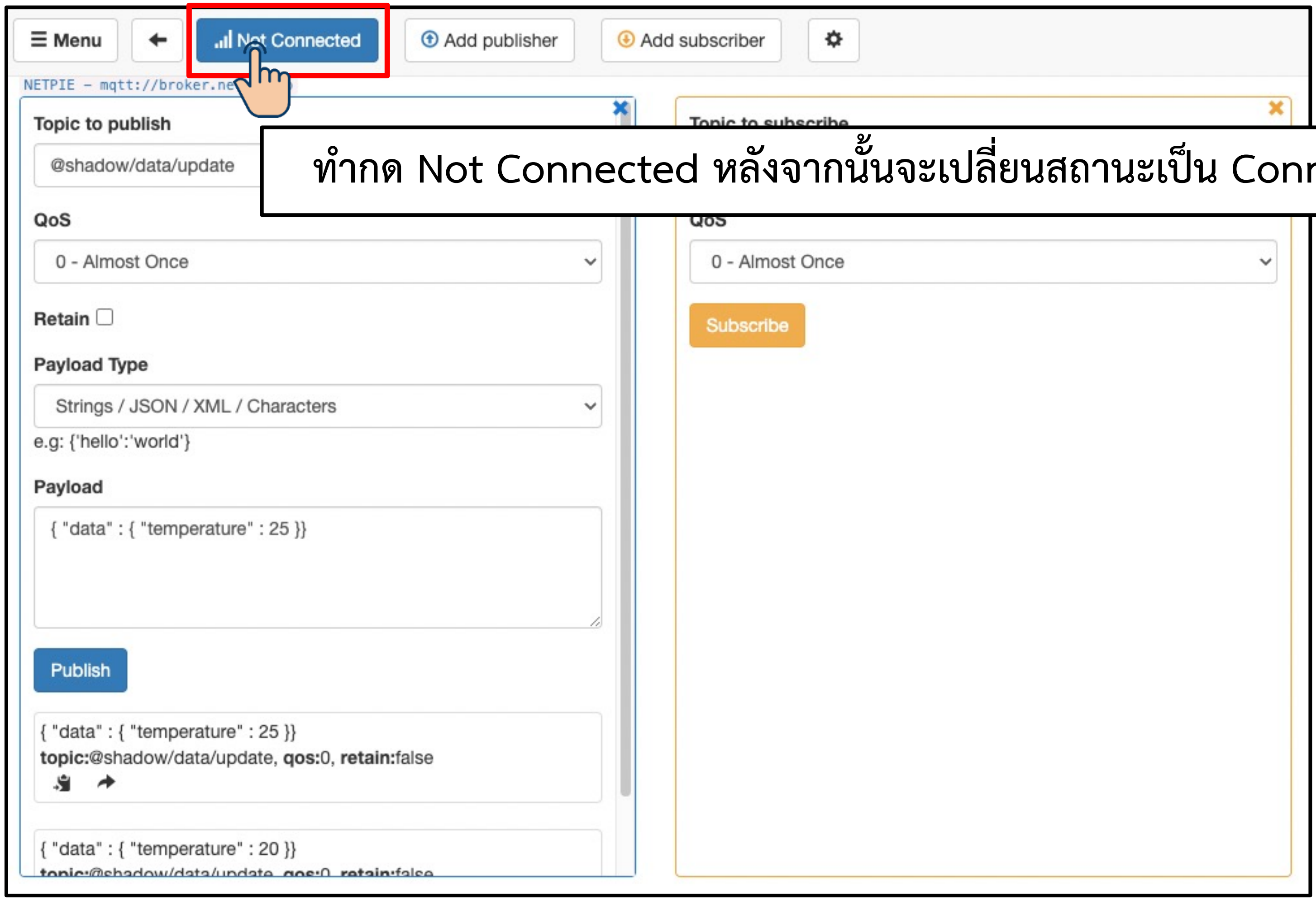
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ข้อความการแจ้งเตือนสถานะที่เปลี่ยนไปถูกส่งมายัง Line

การสร้างการแจ้งเตือนผ่าน Line Notify

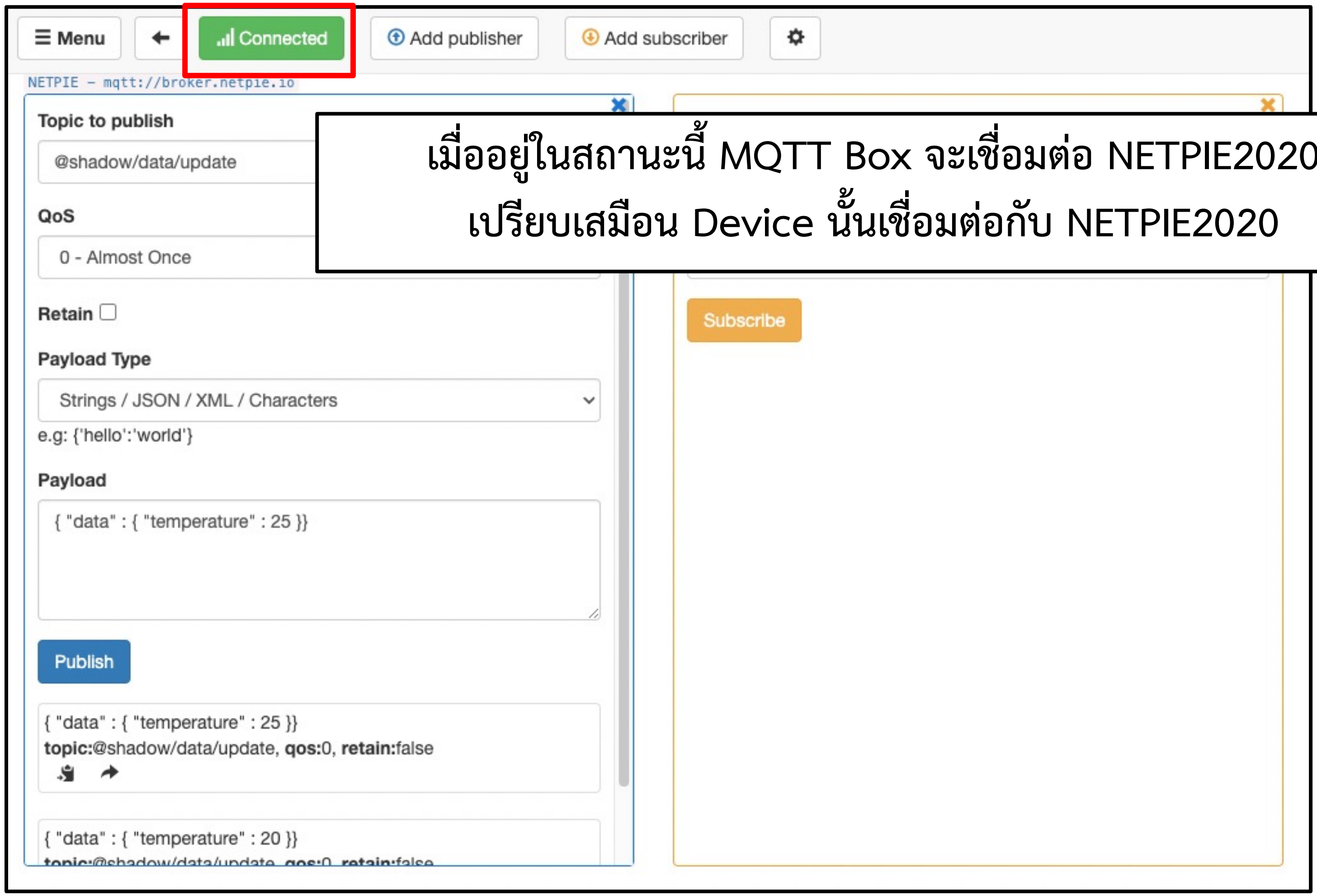
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ทำกด Not Connected หลังจากนั้นจะเปลี่ยนสถานะเป็น Connected

การสร้างการแจ้งเตือนผ่าน Line Notify

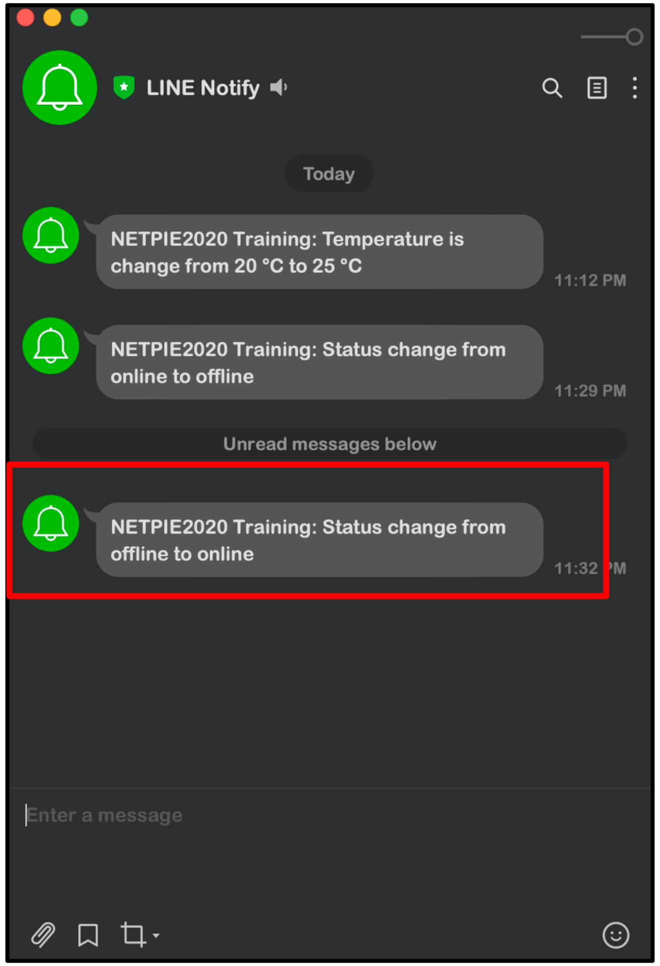
การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



เมื่ออยู่ในสถานะนี้ MQTT Box จะเชื่อมต่อ NETPIE2020
เปรียบเสมือน Device นั้นเชื่อมต่อกับ NETPIE2020

การสร้างการแจ้งเตือนผ่าน Line Notify

การทดลองที่ 2 การสร้าง Device Trigger เพื่อสร้างเงื่อนไขสำหรับเรียกใช้งาน Line Notify



ข้อความการแจ้งเตือนสถานะที่เปลี่ยนไปถูกส่งมายัง Line

ITCS447
Lectures 14-15

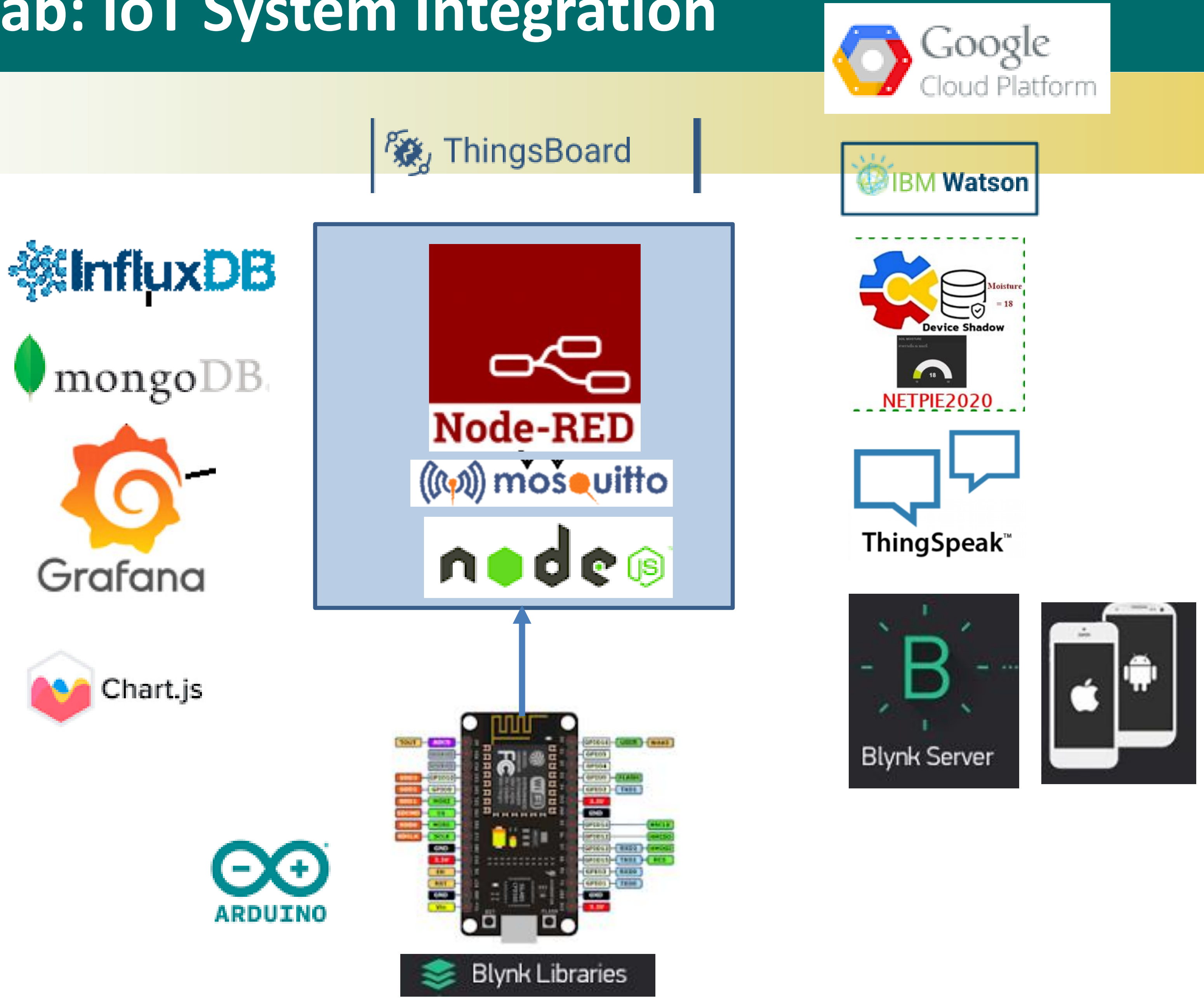
Our Lab: IoT System Integration
IoT Security
Battery Life & Power Management

Asst. Prof. Dr. Thitinan Tantidham

มหาวิทยาลัยมหิดล
Mahidol University

Part 1
IoT System Integration

Our Lab: IoT System Integration

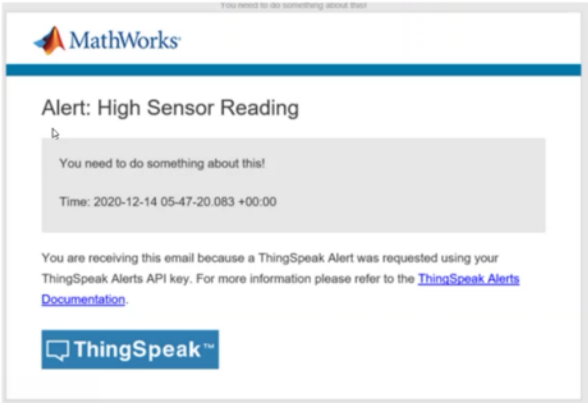


Thingspeak

IoT Workflow

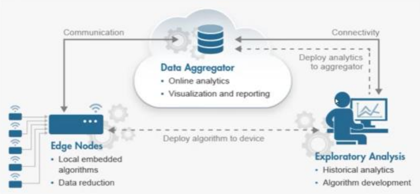
Collect → Analyze → Act

- Acquire data from the field using sensors – often in remote locations!
- Monitor the data in near real time
- Analyses the raw data in the cloud to gain further insight
- React to what the data is telling you
  - Send commands back to the devices
    - e.g., turn on irrigation, turn off motor
  - Send an email or tweet
  - Trigger an HTTP request
    - e.g., to IFTTT or other web services



What is ThingSpeak?

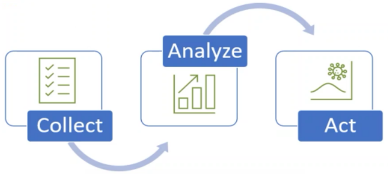
- Online IoT analytics platform
  - Typically used to collect data from sensors ("Things")
  - Provides instant visualization of the data
  - Popular for people experimenting in IoT
  - Free to get started
- Can be used to act on data
  - E.g. Tweet a message when the temperature in your backyard reaches 32 degrees
- Can be used to analyze data
  - MATLAB integration allows users to run scheduled MATLAB code on data coming into ThingSpeak

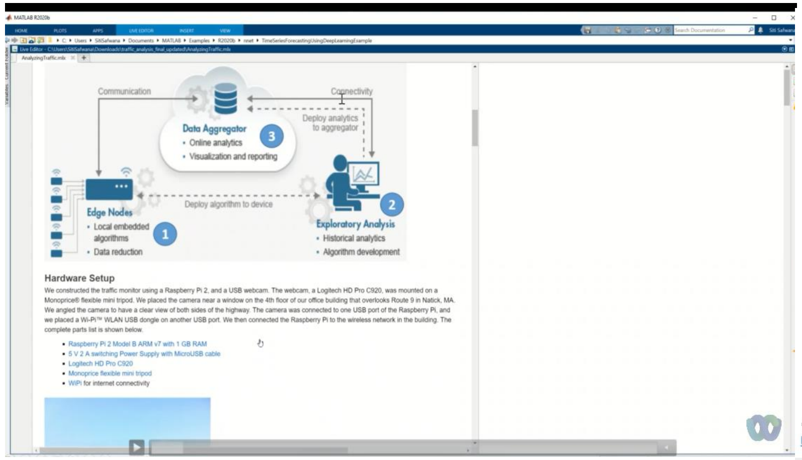


Collect — Send sensor data to the cloud.
Analyze — Analyze and visualize your data.
Act — Trigger a reaction.

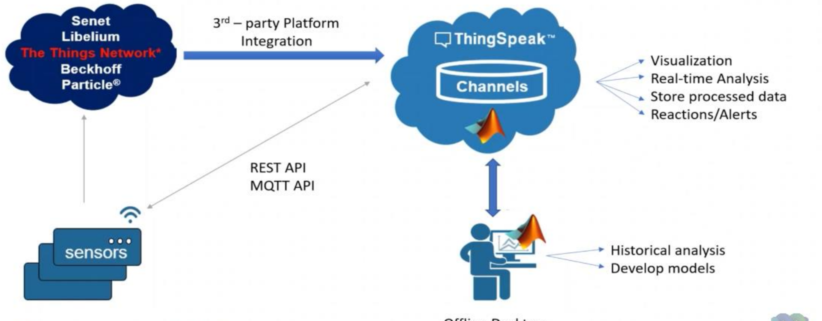
Using ThingSpeak and MATLAB for IoT
Key Takeaways

- ThingSpeak allows you to **collect, analyses and act** on your IoT data
  - **Without setting up the web servers**
  - **Without having expertise in web technology**
- ThingSpeak **natively run MATLAB** for more advanced analysis of your IoT data





How to use ThingSpeak?



*https://www.thethingsnetwork.org/docs/applications/thingspeak
https://www.mathworks.com/help/thingspeak/write_things_network.html

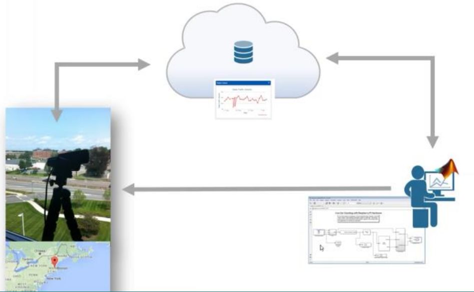
Building a Car Counter

Objectives

- Measure, explore, discover traffic patterns

Solution

- RaspberryPi 2+ webcam
- **Automated deployment of vision algorithms on embedded sensor**



Part 2
IoT Security

Outline

- IoT Security – Related Terminologies
- IoT Characteristics
- Data in IoT
- Data Flow in IoT
- Case Studies
- News
- IoT Attacks
- IoT Reference Model
- IoT Vulnerabilities and attacks
- Edge Computing Security Strategies

https://www.sei.cmu.edu/our-work/projects/display.cfm?customel_datapageid_4050=334957
https://www.iotworldtoday.com/guide/striking-back-an-iot-security-guide-for-critical-infrastructure/
NETACAD Fundamental IoT: IoT Security

IoT Security - Related Terminologies

IoT Security vs Cybersecurity vs CPS

- **IoT Security** is not traditional cybersecurity, but a fusion of cybersecurity with other engineering disciplines.
  - It addresses much more than **mere data, servers, network infrastructure, and information security**. Rather, it includes the direct or distributed monitoring and/or control of **the state of physical systems** connected over the Internet.
  - It requires a unique application for each system and system-of-systems in which IoT devices participate.

- Cybersecurity, if you like that term at all, generally does not address the physical and security aspects of the hardware device or the physical world interactions it can have.

- **Cyber-Physical Systems (CPS)** are a huge. A CPS, comprising connected sensors, actuators, and monitoring/control systems, **does not necessarily have to be connected to the Internet** and still achieves its business objective.

IoT Security – Related Terminologies

Vulnerability - Threat - Attack – Adversary - Countermeasure

- **Vulnerability**: A flaw or weakness in a system's design, implementation, operation, or management that could be exploited to violate the system's confidentiality, integrity, or availability

- **Threat**: Any circumstance or event with the potential to exploit a vulnerability and adversely affect a system through unauthorized access, destruction, disclosure, or modification of data, denial-of-service, etc.

- **Attack**: An intentional assault on system security that derives from an intelligent threat.
  - **Active attacks** attempt to alter system resources or affect their operation
  - **Passive attacks** attempt to learn or make use of information from a system but does not affect that system.

- **Adversary**: An entity that attacks a system or is a threat to a system.

  synonyms: intruder, attacker, cyber attacker, cracker, hacker, etc.

- **Countermeasure**: An action, device, procedure, or technique that meets or opposes (i.e., counters) a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken.

IoT Characteristics

- Heterogeneity of devices – the large number of devices used means high diversity in their calculation and communication capabilities;
- Scalability – addressing, naming, connectivity of a growing number of devices being used every day;
- Wide scale use of wireless data transfer technology – problems connected with overcoming this issue are related to transfer speeds and delays in delivery of data;
- Optimum energy use – the issue of power use is crucial;
- User configuration capabilities – the number as well as the complexity of systems make it necessary to provide mechanisms allowing the users to configure the systems themselves;
- Data management – in IoT it will be crucial to utilize appropriate data models and semantic descriptions of their content, appropriate language and format;
- Privacy protection – because of its close relationship with the real world, IoT technologies must ensure an appropriate level of security and privacy.

Ref: A. Magruk, "The Most Important Aspects of Uncertainty in the Internet of Things Field – Context of Smart Buildings", Procedia Engineering, Elsevier, 2015.

Data in IoT

1: Data Ingestion: IoT devices/sensors collect data from the environment.

The data can be as simple as temperature/humidity or it can be as complex as a full video feed. “Almost 5 quintillion bytes of data produced every day by IoT devices.”

The data needs to be sent to cloud to be analyzed. But it needs a way to get there.

2: Data Transmission: The data is transmitted to the cloud via Gateways (Telemetry Devices).

The gateways use both the cellular as well as satellite communication to transmit the data.

To ensure the data security protocols such as Bluetooth, Sig Fox, LoRa, NB-IoT, ZigBee, COAP, REST, MQTT, etc are used.

3: Data Processing: Once the data gets to the cloud, IoT platform process it.

The processing can be as simple as checking if the temperature is within the acceptable range or it could be very complex, such as using computer vision on video to identify objects.

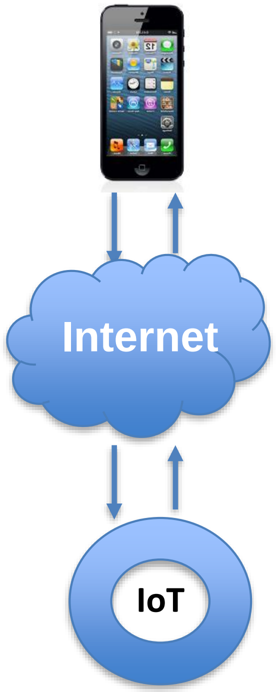
4: Data Visualization: The processed data (information) is made useful to the end-user by providing alerts to the user (E-mails, text, notification).
The user might have an application (interface) that allows him to proactively check-in to the system.

5: Data Analysis and Prediction: To utilize the data collected over the time, data analytics makes use of the historical data to provide actionable insights. Insights helps in predicting the future events that may occur. For example, by analyzing the data, we can predict the possible future malfunctioning of a machinery.

Data Flow in IoT - Operational Models

Security and Privacy Risks

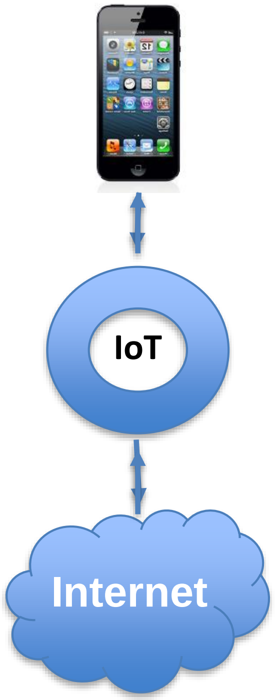
External Server



Eg: Nest Protect Smoke Alarm

https://www.youtube.com/watch?v=81jg6SIxnXM

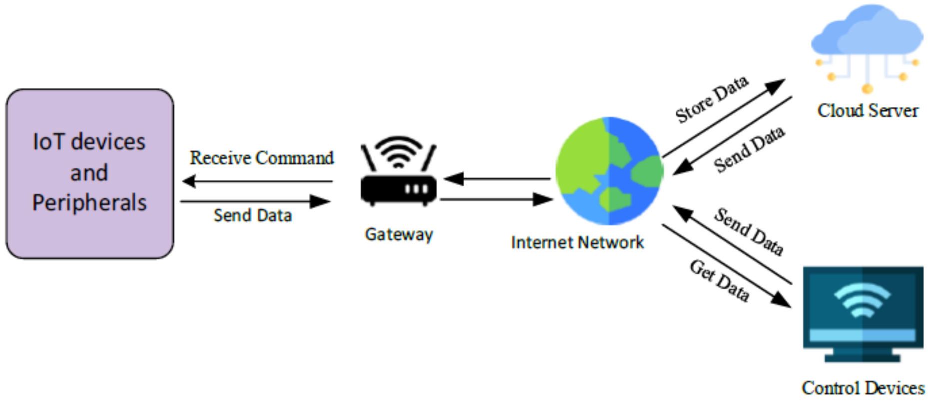
Direct Access



Eg: Philips Hue Lamps/WeMo

Transit

Eg: Fitbit Flex



External Access: the user has no direct interaction with the IoT device. IoT device communicates directly with the external server and the only way for the user to retrieve relevant data, such as current status via this external server.

Direct Access: The IoT device updates the server of the current status. It is also possible to control the IoT device via the web portal provided by the manufacturer.

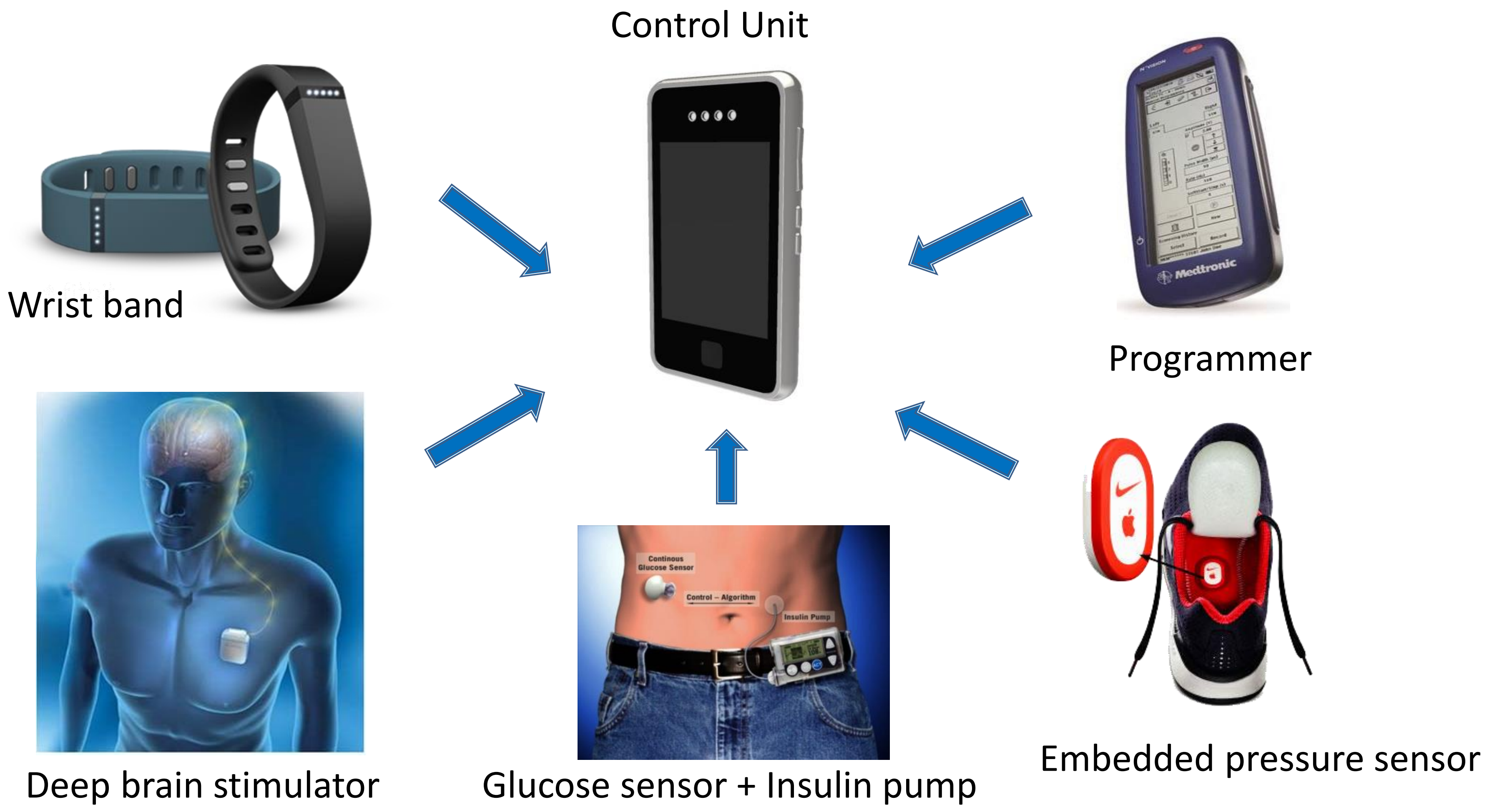
Transit Access: these devices (e.g. Fitbit) do not have Wi-Fi interface, the user's phone is used as a relay/bridge for data exchange.

Ref: Msiddiqi, et.al, "An Experimental Study of Security and Privacy Risks with Emerging Household Appliances, October 2014.

Case Studies: Phillips Hue Lamps

- One of the oldest IoT devices on the market (since 2011).
- Ability to control lights via a smartphone app.
- Highly Customizable and work with a lot of 3rd party services like IFTTT (eg: blink the light if someone sends me a message on facebook)
- Phone talks directly to the hue bridge and bridge then relays appropriate commands to the lights using zigbee.
  All Communications between the phone and the bridge are **in plain text**.

Case Studies: E-Health Applications



News in the IoT Security Fails

- “Lack of Trust in Internet Privacy and Security May Deter Economic and Other Online Activities ”, NTIA May 2016.
- How long will consumers put up with the IoT's failures? – IoT support panel, CES 2016.
- IoT “plug and pray” all over again, says security consultant David Alexander, PA Consulting, CRESTcon & IISP 2016.
- Three quarters of UK’s information security professionals think IoT device manufacturers aren’t implementing enough security on their products and 73% said there’s a general lack of industry standards – ISACA 2015 poll
- 72% of Americans see cyberattacks as a major threat, coming 2nd after ISIS – Pew Research poll, April 2016.
- “All of the potential weaknesses that could afflict IoT systems, such as authentication and traffic encryption, are already well known to the security industry... ”, Insecurity in the Internet of Things, Symantec, March 2015

https://cisomag.eccouncil.org/10-iot-security-incidents-that-make-you-feel-less-secure/

https://portswigger.net/daily-swig/iot

News: Case Studies (1/2)

The IoT Risk and Security Awareness

- Target’s Heating and Cooling System
  - Hackers gained access through HVAC account, and were able to install card skimming s/w on POS terminals
- Wink’s IoT Hubs (http://www.wink.com/)
  - Consumers found their devices bricked when the Hub security certificate unexpectedly expired
- Insteon connected homes
  - Reporter able to turn lights on and off whilst chatting with home owners over the phone
- Home routers
  - Open to man in the middle attacks when people use default or easy to guess passwords
- Spammy refrigerators
  - Default passwords allowed attacker to use connected refrigerators as part of a `bot net
- TrendNet’s nanny cams
  - Easy remote access once you have the camera’s IP address
- Samsung’s smart TVs
  - Easy to commandeer to view people’s living rooms
- Nest thermostat
  - Easy to hack if you can get physical access for a few minutes

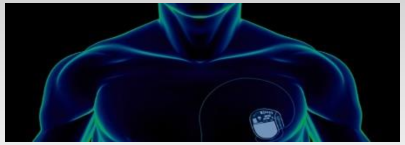
https://www.w3.org/Talks/2016/0614-iot-security.pdf
https://www.forbes.com/sites/kashmirhill/2013/07/26/smart-homes-hack/?sh=7dedc710e426

News: Case Studies (2/2)

The IoT Risk and Security Awareness

Vehicle Hacking
http://bit.ly/1s0m4Hv
http://bit.ly/1TOt2h5

Global Positioning System Spoofing
https://www.youtube.com/watch?v=y4pr5_ea5hw



Healthcare Device & Information Hacking
http://bit.ly/1EJnTjv

Industrial Hacking

Smart Home Hacking
https://www.youtube.com/watch?v=-JVrFI138kE

National Transportation Safety Board (NTSB) Connected-Car Mandate
https://ecfsapi.fcc.gov/file/103101033822776/Final_Security%20Considerations%20for%20Connected%20Vehicles%20and%20DSRC_3_19_2017.pdf

- Breaches of privacy
- Cybercrime
- Physical safety in the home, across the city and within businesses
- Threats to national infrastructure
- Looming risks of cyberwar

https://www.youtube.com/watch?v=iRQPfISsG9k

https://www.youtube.com/watch?v=Fki7MCRWgdo

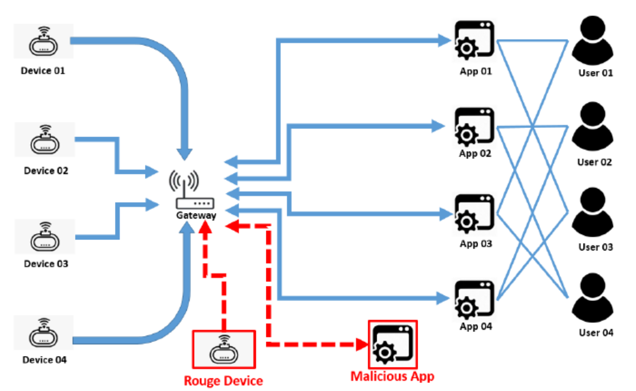
IoT Attacks

- Default, weak, and hardcoded credentials
- Difficult to update firmware and OS
- Lack of vendor support for repairing vulnerabilities
- Vulnerable web interfaces (SQL injection, XSS)
- Coding errors (buffer overflow)
- Clear text protocols and unnecessary open ports
- DoS / DDoS
- Physical theft and tampering

Types of Attacks

- Wireless Reconnaissance and Mapping
  - Similar to the war dialing days of old where hackers scanned through telephone switching networks to identify electronic modems
  - Network scanning tool such as Nmap is commonly utilized by hackers to gather intelligence about hosts, subnets, ports, and protocols in networks, similar paradigms are being used against IoT devices
- Security Protocol Attacks
  - Many security protocols can sustain attacks against vulnerabilities introduced either in the protocol design (specification), implementation and even configuration stages (in which different, viable protocol options are set).
  - As an example, researchers found while testing a ZigBee-based consumer IoT implementation that the protocol was designed for easy setup and usage but lacked configuration possibilities for security and performed vulnerable device pairing procedures.
- Physical Security Attacks
  - Physical security attacks include those in which the attacker(s) physically penetrate the enclosure of a host, embedded device, or other type of IoT computing platform to gain access to its processor, memory devices, and other sensitive components.
- Application Security Attacks
  - IoT devices and connections can be exploited through attacks against application endpoints. Application endpoints include web servers as well as mobile device applications (for example, iPhone, Android) that have a role in controlling the device.

Common IoT Attacks



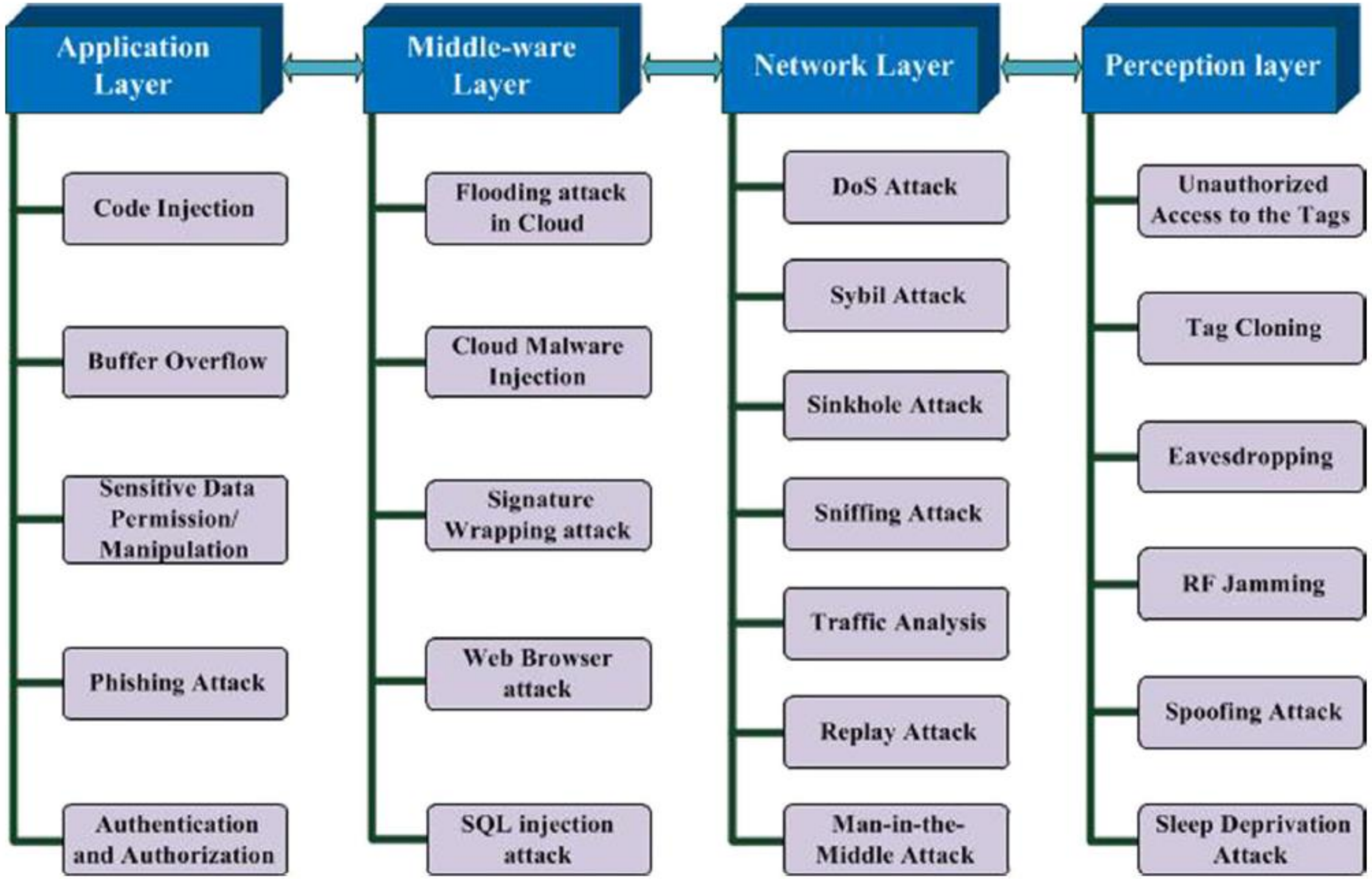
- Lack of compliance on the part of IoT manufacturers (e.g.: fitness trackers, smart refrigerators, etc.)
- Lack of user knowledge and awareness
- IoT security problems in device update management
- Lack of physical hardening

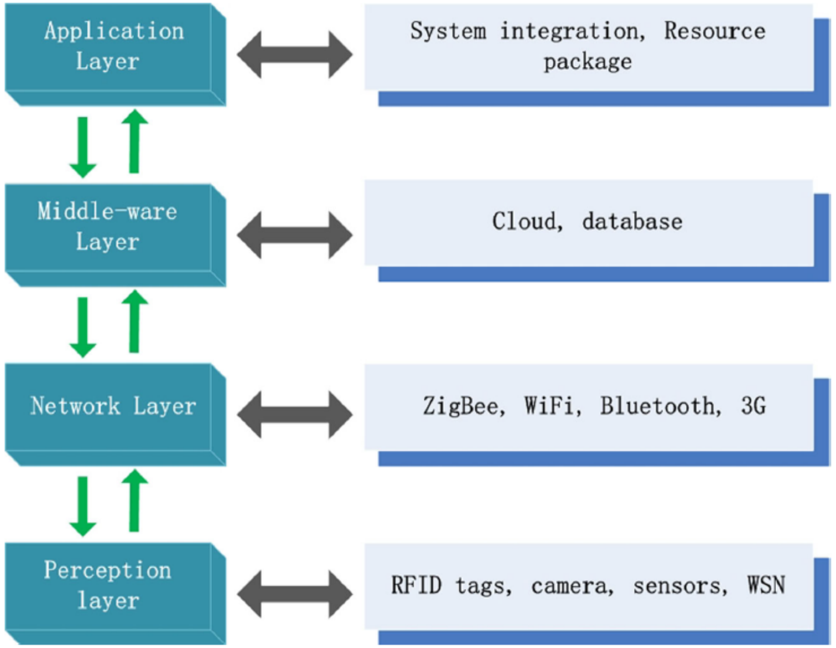
- Botnet attacks: IoT devices are highly vulnerable to malware attacks. A victim's system can be remotely take control and confidential data can be storlen.
- Industrial espionage and eavesdropping: Attackers can sniff the traffic generated by IoT data flow to gather users' critical information (usernames/passwords, identifiers, other useful data) by setting similar IoT devices.
- Highjacking IoT devices (ransomware does not destroy sensitive files but blocks access to them by way of encryption)
- Data integrity risks of IoT security in healthcare
- Rogue IoT devices (without authorization)
- Cryptomining with IoT Bots (Mining cryptocurrency demands colossal CPU and GPU resources)
- Malicious Data Injection: False sensor data injection is a form of attack in which the sensor data used in IoT applications are forged or modified for malicious purposes.
- Sybil Attack: The malicious nodes in this can have multiple identities of a genuine node by either impersonating it or with a fake identity through duplication. One such malicious node may have several identities simultaneously or at different instances or an attacker can be in more than one place at once.
- Disclosure of Critical Information: Sensors used in IoT gadgets can disclose sensitive information such as passwords, secret keys, credit card credentials, and so on.
- Sinkhole attack: all traffic is lured from an area through a compromised node, where selective forwarding can follow with the attacker deciding what data to allow through.
- Wormhole attack – an attacker tunnels messages received in one part of the network over a low latency link and replays them in a different part.
- HELLO flood – here the attacker causes every node to mark it as their parent. Most nodes will be out of range and this causes a lot of packets to be lost. Routing loops can be set up via spoofing routing updates, with two nodes being attacked and redirecting packets to each other.
- Acknowledgement spoofing - used for a selective for-warding attack, where an attacker strengthens/weakens networks links so packets are lost from a node.
- Denial of Service (DoS) attacks – jamming radio signals, using malicious nodes to refuse to route messages or redirect them to unwanted locations.
- MITM(Man-in-the-Middle): a system that listens on traffic between smart device and a gateway.
- Ransomware: a hacker uses malware to encrypt data that may be required for business operations. An attacker will decrypt critical data only after receiving a ransom.

Ref: https://www.intellectsoft.net/blog/biggest-iot-security-issues/

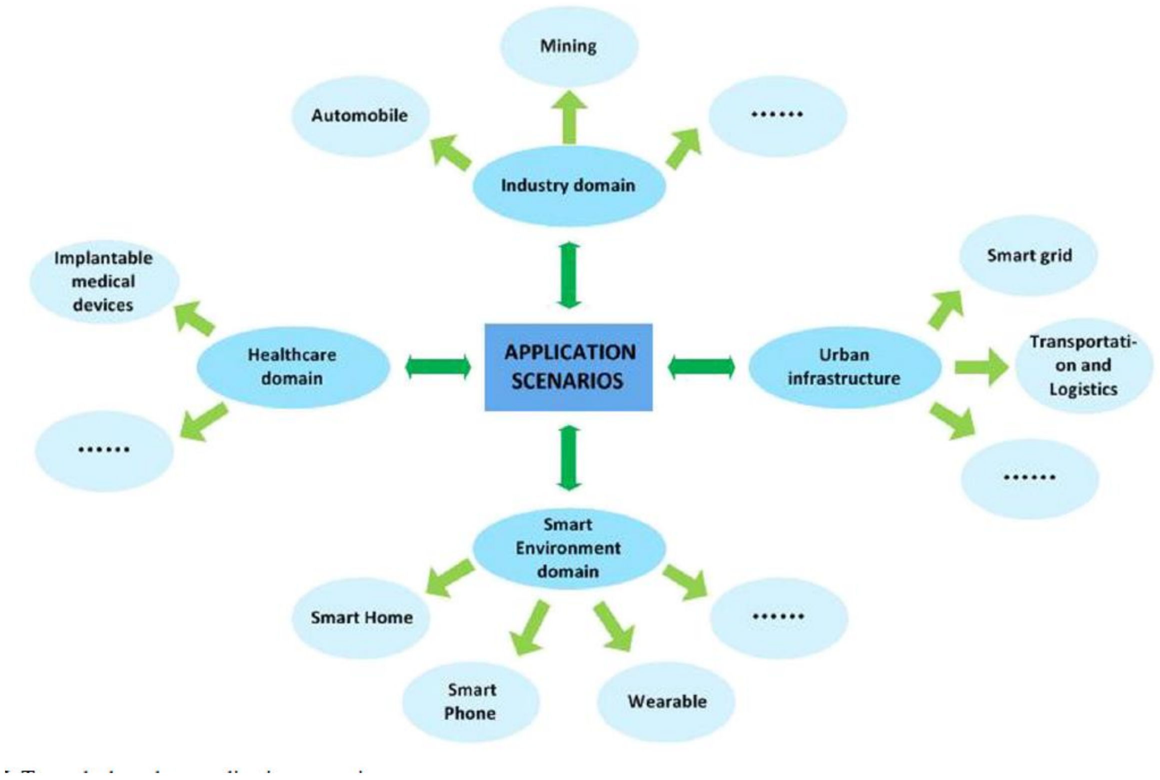
IoT Attacks

Types of Attacks based Architecture

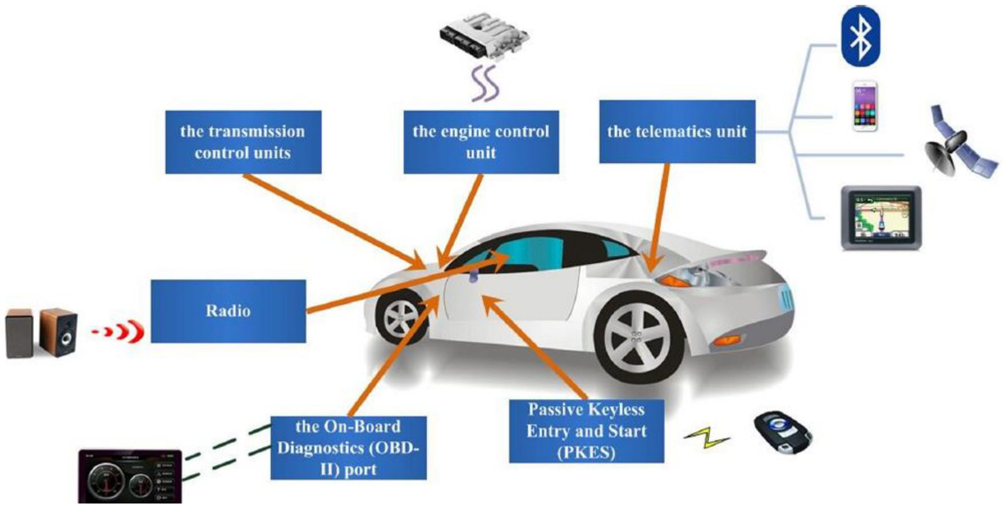




Types of Attacks based Application Scenarios



Reverse engineering can also be used to "crack" software and media to remove their copy protection, or to create a (possibly improved) copy or even a knockoff; this is usually the goal of a competitor

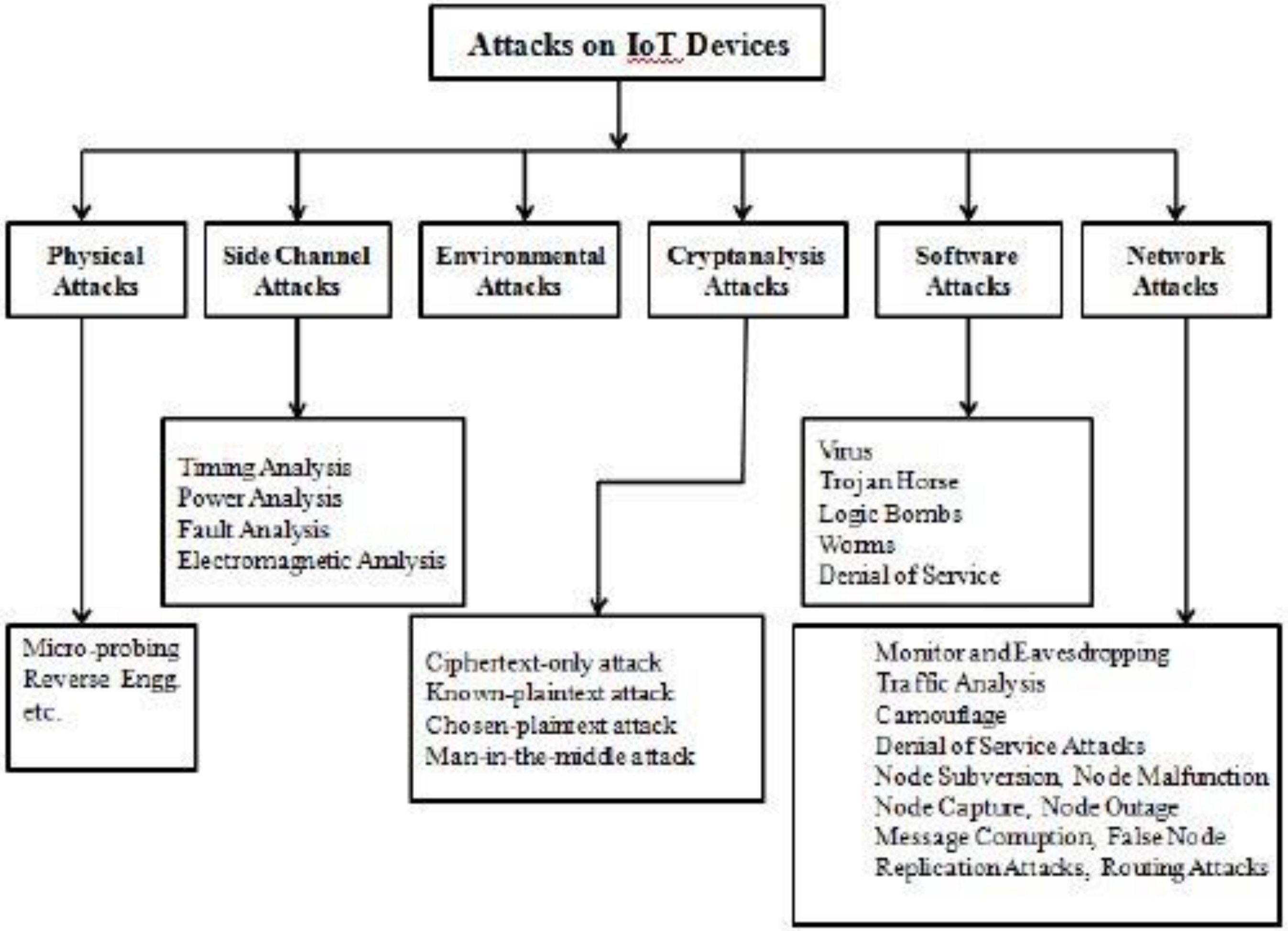


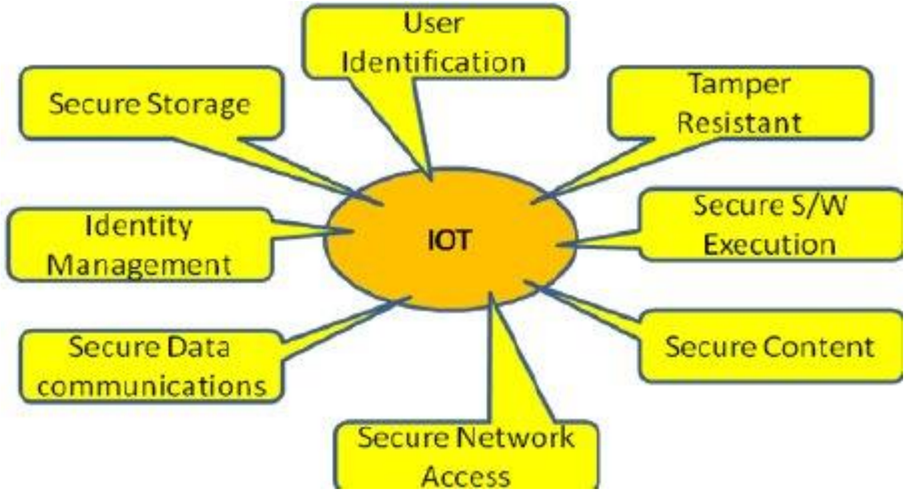
Types of Attacks based on Sensor Vulnerabilities

Sensor Type	Sensor	Vulnerabilities
Motion Sensors	Accelerometer Gyroscope Linear Acceleration Sensor	Task Inference False Data Injection Malware Transmission
Environmental Sensors	Light Sensor Proximity Sensor Air Pressure Sensor Audio Sensor Temperature Sensor Soil Moisture sensor Noise Sensor	Eavesdropping Task Inference Smudge Attack False Data Injection Transferring Malware DoS Information Leakage
Position Sensors	GPS Magnetic Sensor	Location Inference Eavesdropping False Data Injection

IoT Attacks

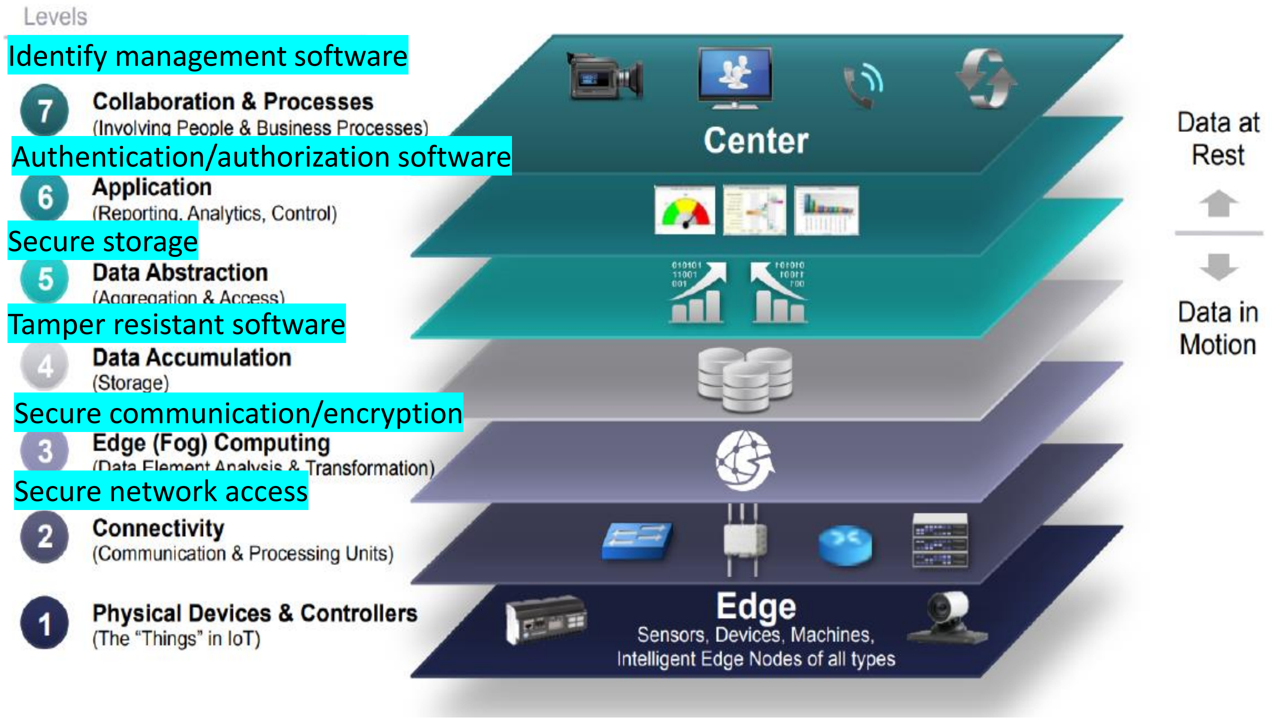
Attack on IoT Devices

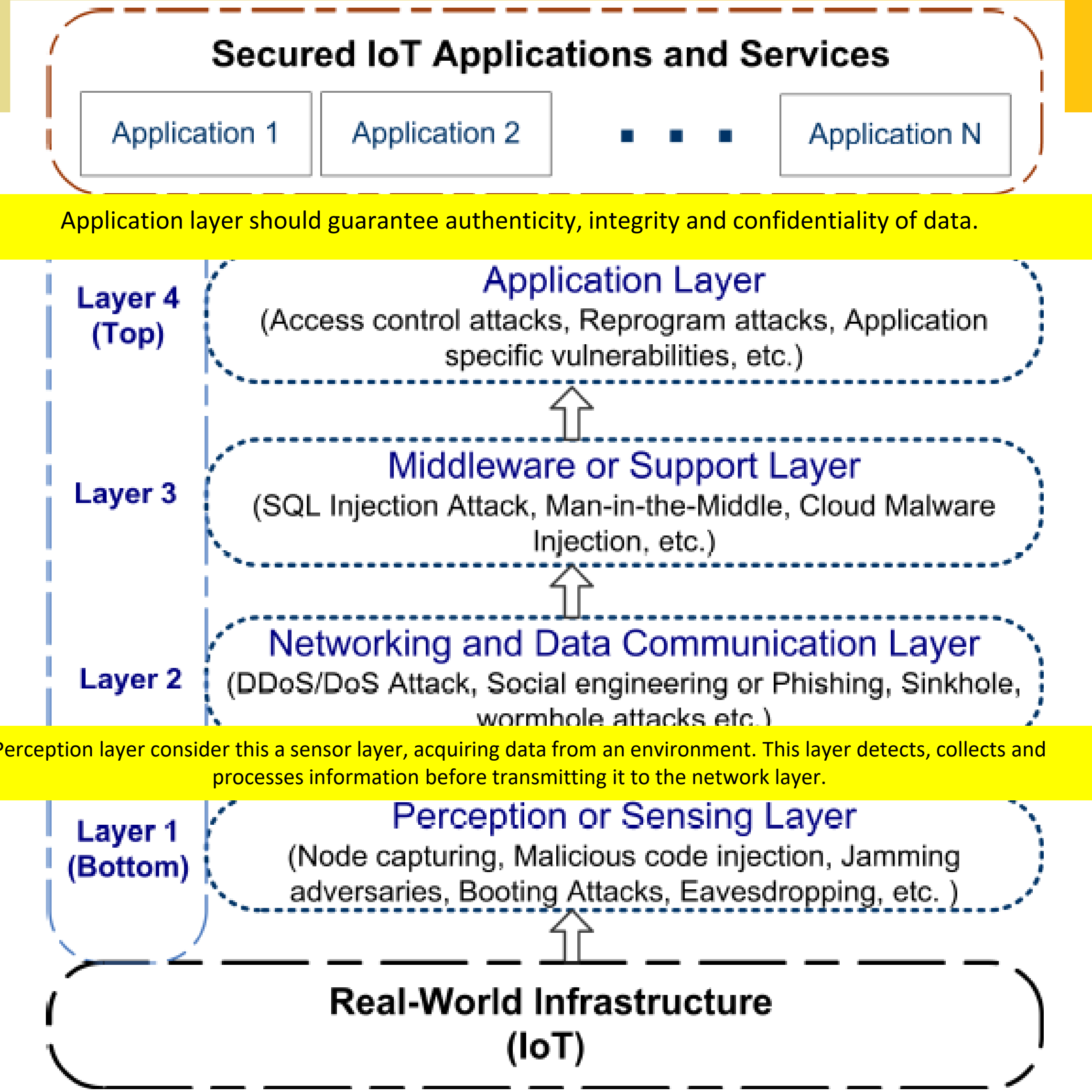




IoT Attack based on IoT Reference Model

There is no single standard and unified IoT layer model.



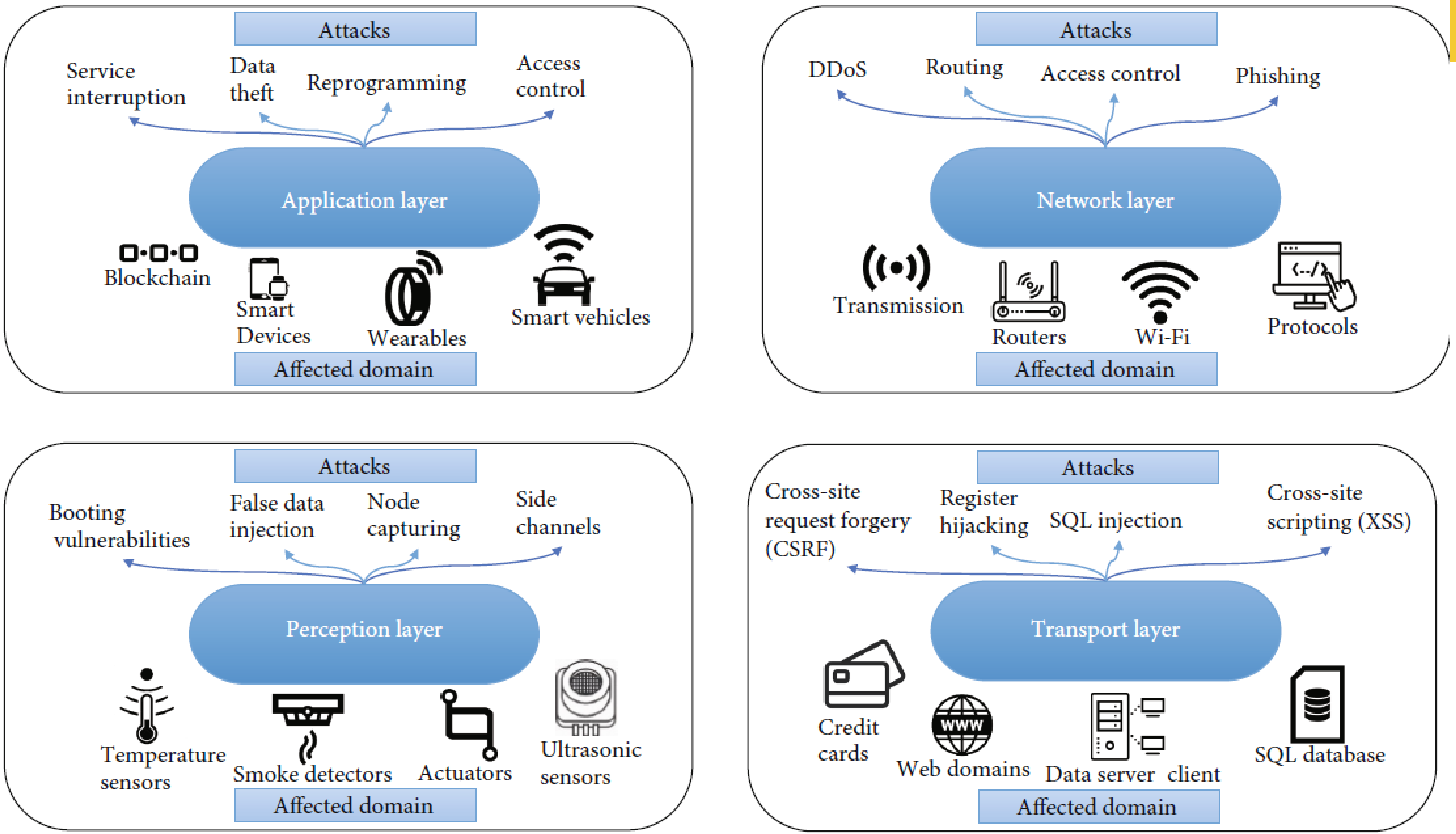


Application layer should guarantee authenticity, integrity and confidentiality of data.

Perception layer consider this a sensor layer, acquiring data from an environment. This layer detects, collects and processes information before transmitting it to the network layer.

S. Millar, "IoT Security Challenges and Mitigations: An Introduction", 2021.

IoT Attack based on IoT Reference Model



R., S. Kataria, et.al, “Threats and Corrective Measures for IoT Security with Observance of Cybercrime: A Survey”, 2021.

Taxonomy of Threats in IoT

- Perception Layer: Eavesdropping, Battery drainage, Hardware failure, Malicious data injection, Sybil threat, Disclosure of critical information, Device comprise, Node cloning
- Abstraction Layer: Node replication, Illegal access, Device compromise, MITM, Eavesdropping, Spoofing, Insertion of rogue devices, Information theft, Threats to communication protocols
- Network Layer: Illegal access, MITM, Eavesdropping, Spoofing, Fragmentation threat, Hello flood, Network intrusion, Device compromise, Node replication, Insertion of rogue devices
- Transport Layer: Jamming, Eavesdropping, False data injection, Unfair access, Congestion, Hello flood
- Computing Layer: Malicious attack, SQL injection, Data integrity, Virtualization, Software modification, Illegal access, Identity theft
- Operation Layer: Fake information, Badmouthing, Unauthorized access, User privacy compromise, Stealing user critical information
- Application Layer: Malicious code, Software modification, Data tampering, SQL injection, Disclosure of critical information, Cross site script, Identity theft

Taxonomy of Attacks in IoT

- Perception Layer: Node capture attack, Side channel attack, Tag cloning, RF jamming, node injection attack, Node tampering, Physical damage, Exhaustion attack, Node outage, Battery drainage attack
- Abstraction Layer: Data manipulation attack, Device tampering attack, Tag cloning, MITM attack, Dos attack, DDoS attack, Side channel attack, Traffic analysis, Sleep deprivation attack
- Network Layer: Eavesdropping attack, Hello flood attack, Sinkhole attack, Sybil attack, Clone ID attack, Selective forwarding attack, Black hole attack, Wormhole attack, Traffic attack, RPL exploit
- Transport Layer: DoS attack, DDoS attack, Side channel attack, Desynchronization, MQTT exploit, Session hijacking, SYN-flooding attack, Timing attack, Unfairness attack
- Computing Layer: Flooding attack in cloud/edge, Malware injection attack, Access attack, False data injection, Path based DoS attack, Hole attack, Exhaustion attack, Edge/Cloud outage, Signature wrapping, Storage attack
- Operation Layer: Man-in-the-middle attack, Secure on-boarding attack, Firmware attack, Software attack, Illegal intervention, End-to-end encryption attack, Interrogation attack, DoS attack
- Application Layer: Virus attack, Malware attack, Spyware attack, Flooding, Spoofing, Code injection, Intersection, Message forging DDoS attack, Brute force attack.

IoT Device/System/Software Attack Tools

- Kali Linux
- Tools (all available in Kali distro):
  – *wireshark:* www.wireshark.org
  – *nmap / zenmap:* https://nmap.org/, https://nmap.org/zenmap/
- *aircrack-ng / fern*
  - https://www.aircrack-ng.org/
  - https://tuxdiary.com/2015/08/30/fern-wifi-cracker/
- metasploit / armitage
  - https://www.metasploit.com/
  - http://www.fastandeasyhacking.com/
- OWSAP Zed
  - https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project
- Nikto
  - http://sectools.org/tool/nikto/
- Sqlmap
  - http://sqlmap.org/
- Social Engineer Toolkit
  - https://www.trustedsec.com/social-engineer-toolkit/
- Maltego
  - https://www.paterva.com/web7/
- Shodan (the search engine website scans the Internet for publicly accessible devices.

  –shodan.io

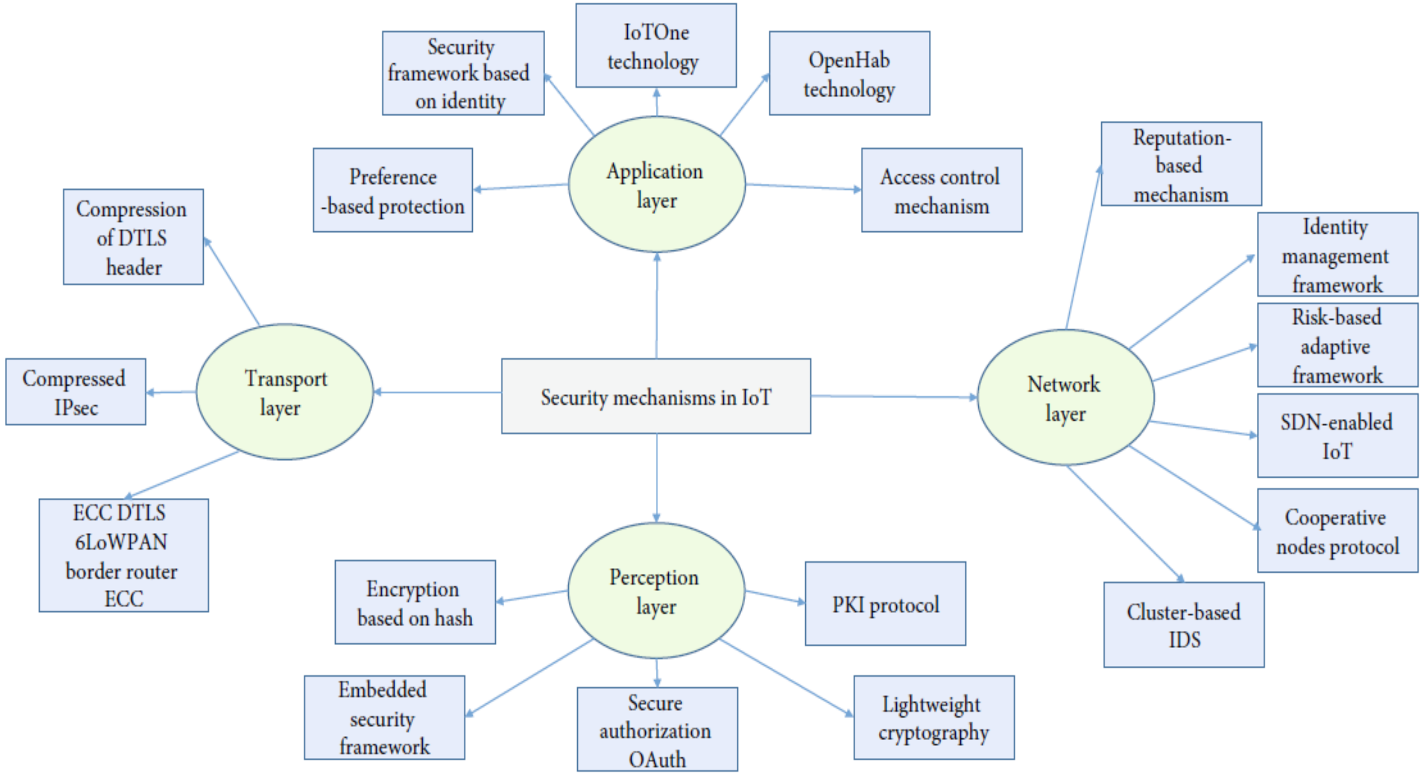
Red teams are offensive security professionals who are experts in attacking systems and breaking into defenses.

Blue teams are defensive security professionals responsible for maintaining internal network defenses against all cyber attacks and threats

IoT Device Attack Tools

Software	OS/Support	Features	Sources
E3 Universal	Window, Linux, macOS, iOS	IoT analysis, cloud data imaging, and analysis, registry analysis, email investigation, JTAG, and chip dump processing	https://paraben.com/digital-forensic-tools-6/
WireShark	Windows, Linux, macOS, Solaris	VoIP, GUI, offline analysis, WAN/LAN analyzer	https://www.wireshark.org/
Autopsy	Windows, Linux, macOS Android	Registry analysis, LNK file analysis, timeline analysis, file type detection, email analysis	https://www.sleuthkit.org/autopsy/
Paladin	Linux	Device cloning support for many forensic image formats: E01, Ex01, RAW, VHD, AFF, disk manager, and automatic logging	https://sumuri.com/software/paladin/
Dumpzilla	Unix, Windows	Forensic information extraction from Firefox, SeaMonkey browsers including cookies, bookmarks, web forms, SSL certificates, browser-saved passwords	https://tools.kali.org/forensics/dumpzilla
SIFT (SANS investigative forensic toolkit)	Linux	File system support, different evidence image format support, rapid scripting, and analysis	https://digital-forensics.sans.org/community/downloads
Toolsley	Web based	File repairing, text encoding, file identification, file signature verification, binary inspection, CRC tool	https://www.toolsley.com/
NetworkMiner	Windows, Linux, macOS X, FreeBSD	Live sniffing, OS fingerprinting, Geo IP localization, DNS whitelisting, audio extraction and playback of VoIP calls, PCAP and PcapNG file parsing	https://sectools.org/tool/networkminer/
Elcomsoft	Windows, macOS, iOS	Password recovery, cloud explorer, disk decryption, wireless security auditor	https://www.elcomsoft.co.uk/
Belkasoft X	Windows macOS, Linux, iOS, Android, Blackberry	E01/DD imaging, Hash set analysis, registry viewer, plist viewer, artifacts viewer, SQLite viewer	https://belkasoft.com/

Existing Security Mechanisms



Network layer	Attacks	Defenses
Physical	Jamming	Spread-spectrum, priority messages, lower duty cycle, region mapping, mode change
	Tampering	Tampering-proofing, hiding
Link	Collision	Error-correcting code
	Exhaustion	Rate limitation
	Unfairness	Small frames
Network and routing	Neglect and greed	Redundancy, probing
	Homing	Encryption
	Misdirection	Egress filtering, authorization, monitoring
	Black holes	Authorization, monitoring, redundancy
Transport	Flooding	Client puzzles
	Desynchronization	Authentication

Challenges for the IoT Security

According to IoT Architecture and Components

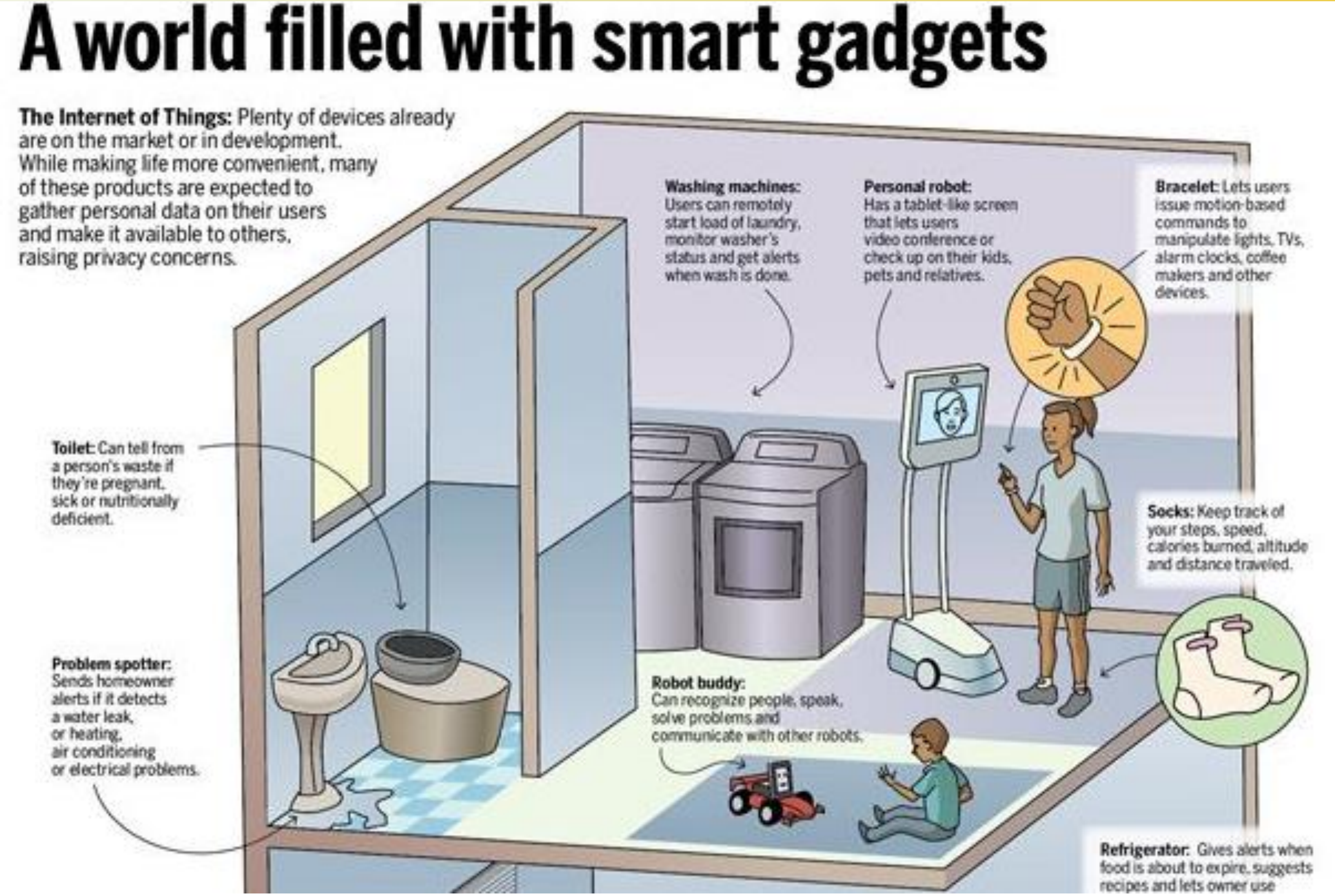
- Diversity of devices, capabilities, standards, programming environment
- Lots of these devices store personal data – e.g. fitness/health, home security, etc.
- Application driven field – innovation comes from non-tech people
  – First to market may mean no/little security
- Potential threats/attacks a guess work.

Challenges for the IoT Security

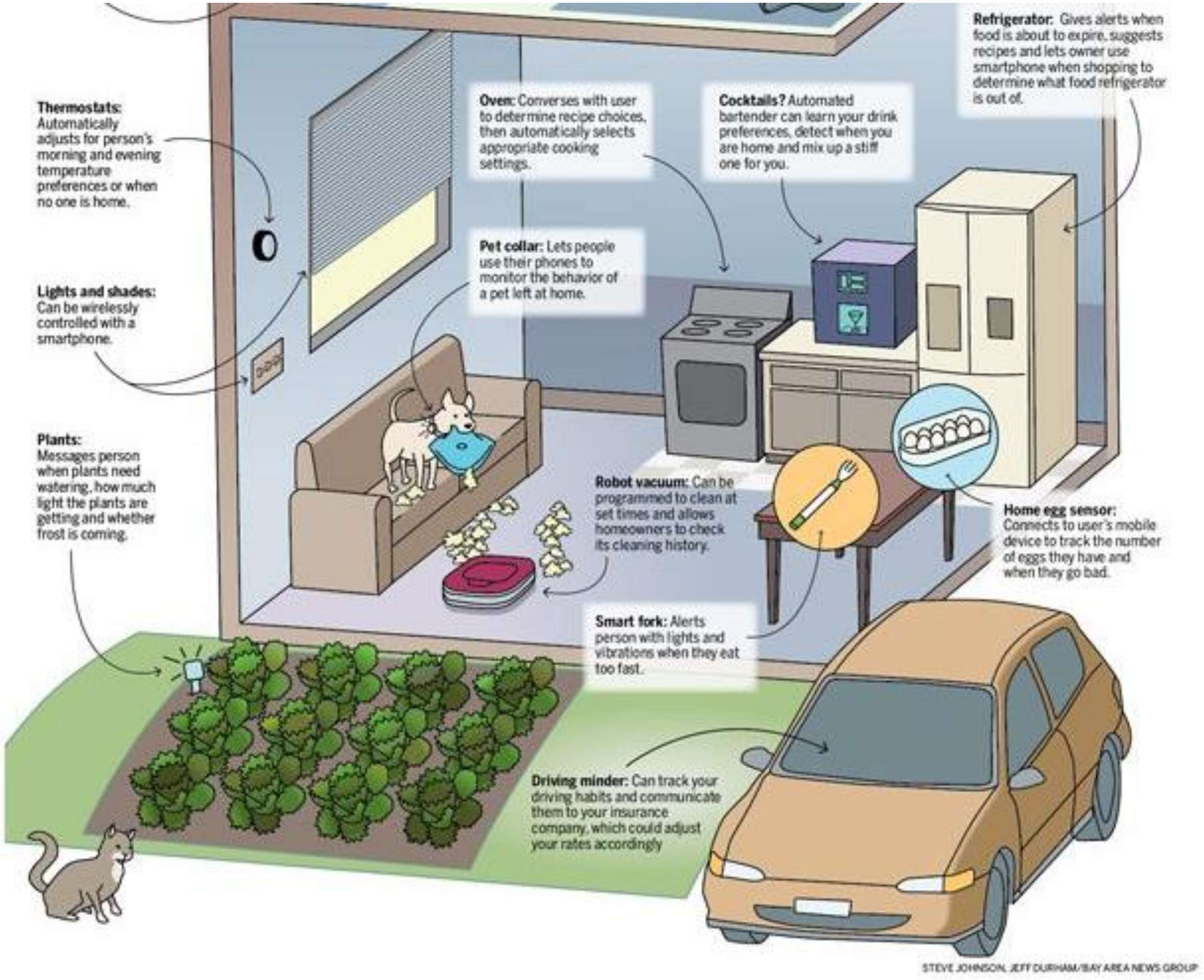
Unique Challenges

- IoT relies on microcontrollers with limited memory and computational power
  - This often makes it impractical to implement approaches designed for powerful computers
  - This in turn requires constrained IoT devices to be hidden behind secure gateways
- Threats based upon gaining physical access to IoT devices
- How to bootstrap trust and security, and ways that this can unravel
- Evolving technology
  - More powerful Systems on a Chip (SOC) embedding hardware security support
  - Ecliptic Curve Cryptography with reduced computational demands
- Anything that is exposed to the Internet must be securely software upgradable
- User experience must be good enough to avoid becoming a weak link in the chain
- The necessity of keeping up to date with security best practices

How Secure are All These Devices? (1/2)



How Secure are All These Devices? (2/2)



Challenges for the IoT Security

IoT Devices (Weakness)

- Devices are not reachable
  - Most of the time a device is not connected
- Devices can be lost and stolen
  - Makes security difficult when the device is not connected
- Devices are not crypto-engines
  - Strong security difficult without processing power
- Devices have finite life
  - Credentials need to be tied to lifetime
- Devices are transportable
  - Will cross borders
- Devices need to be recognised by many readers
  - What data is released to what reader?

Challenges for IoT the Security

IoT and Big Data

- Lots of sensors will generate a vast amount of data
  - API Research estimated 200 exabytes in 2014 and 1.6 zettabytes in 2020
  - 90% is currently processed locally, although this varies by domain
- This creates a greater volume of sensitive data, creating a greater risk of
  - Data Confidentiality
  - Data and identity theft
  - Device manipulation
  - Data falsification
  - Data ownership
  - Data lifecycle management
  - Data Publication
  - IP theft, server/network manipulation, etc.
- Privacy Preserving Data Correlation
  - Personal and population privacy
  - Privacy enhancing techniques
  - Data service monetization
- Impact of introduction of data consolidation and analytics at network edge
  - Cisco, HPE and others
  - App platforms in the cloud or at the network edge will be targets for attacks

Privacy Breaches

IoTs and Big Data

CNNMoney
A Service of CNN, Fortune & Money
FORTUNE | Money

Home | Video | Business News | Markets | My Portfolio | Investing | Economy | Tech | Personal Finance

All Latest Stories | Companies | World | The Buzz | Fortune 500 | Interactive | Video

Target: 40 million credit cards compromised

By CNNMoney Staff @CNNMoney December 19, 2013: 4:41 PM ET

Recommend 62k

iHealthBeat
Reporting Technology's Impact on Health Care
A service of the California HealthCare Foundation

August 02, 2012 - Topic: Privacy and Security

Nearly 21M Affected by Large Health Data Breaches Since 2009

Nearly 21 million individuals have been affected by large health data breaches since HHS' Office for Civil Rights began publicly reporting such incidents in September 2009, *Modern Healthcare* reports (Conn, *Modern Healthcare*, 8/1).

THE WALL STREET JOURNAL.
WSJ.com

BUSINESS | June 9, 2011

Hacking At Citi Is Latest Data Scare

By VICTORIA MCGRANE And RANDALL SMITH

Citigroup Inc. plans to send replacement credit cards to about 100,000 North American customers after its systems were breached by a hacking attack affecting about 200,000 accounts.

Citi said on Thursday that the hacked accounts amounted to about 1% of its 21 million North American card customers and that it has referred the incident to law enforcement. The bank said it is contacting affected customers and has implemented procedures to prevent a recurrence.

Reports: 77 Million PlayStation Network Accounts Compromised

By Nick Mediati, PCWorld

To say that Sony has a mess on its hands may be the understatement of the year. On Tuesday, Sony posted a blog entry that provides some more information on the recent PlayStation Network hack. According to Sony, hackers obtained users' names, addresses, e-mail addresses birthdates, and account login and password, and may have also taken users' security questions and answers. If you set up a sub-account for your child, that information may also be in hackers' hands.

The IoT Security Awareness Issues

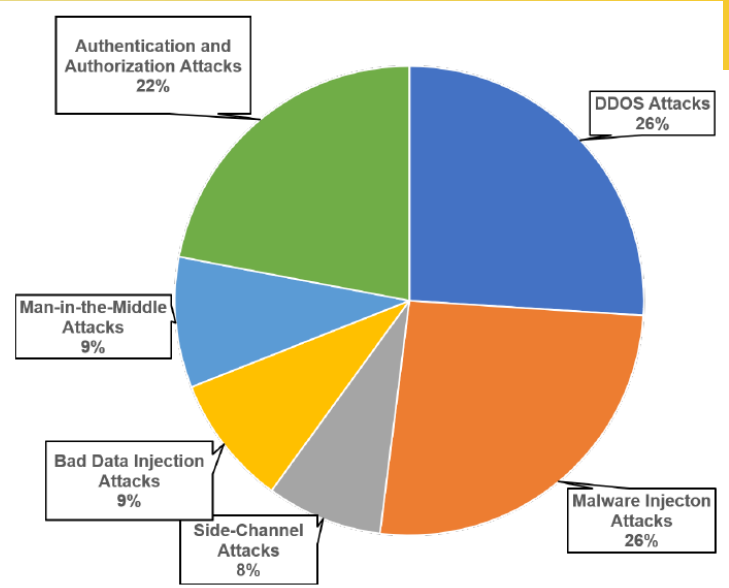
- February 2016 IEEE Experts in Technology & Policy meeting identified a number of areas where we can start:
  - Education & Ethics
  - Data localization
  - Identity management
  - Technology policy development process
  - Autonomy
  - Accountability
  - Tradeoff adjudication
  - Solutions roadmap creation
  - End-to-end security/privacy by design

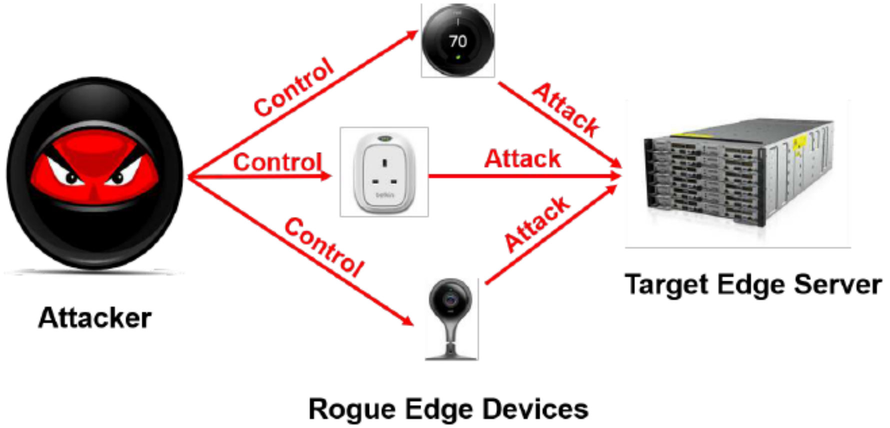
IoT Security Foundation
https://www.iotsecurityfoundation.org/iot-security-resources/

Edge Computing Security

Best Practices

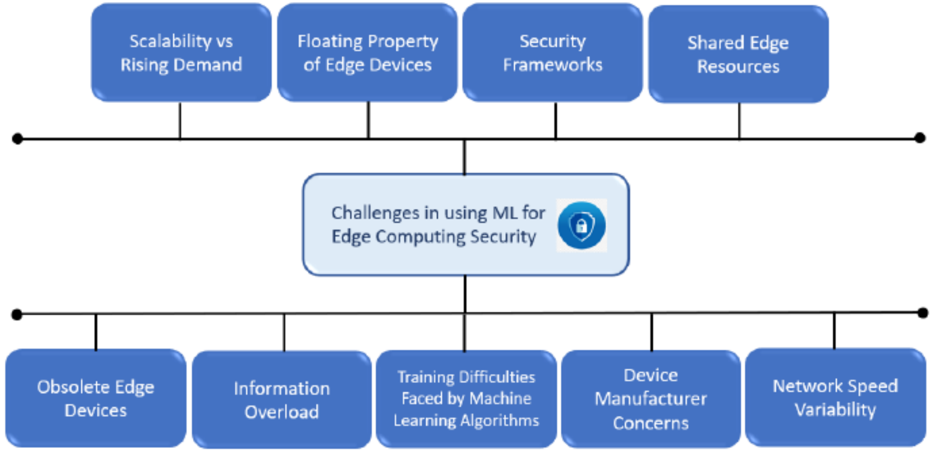
- Use access control and surveillance to enhance the physical security at the edge.
- Control edge configuration and operation from central IT operations.
- Establish audit procedures to control data and application hosting changes at the edge.
- Apply the highest level of network security possible between devices/users and edge facilities.
- Treat the edge as a part of the public cloud portion of your IT operation.
- Monitor and log all edge activity, particularly activity relating to operations and configuration.





https://www.techtarget.com/iotagenda/tip/Edge-computing-security-risks-and-how-to-overcome-them

ML Assisted Security and Privacy Provisioning for Edge Computing



Attack	Reference	Security Breach Technique	ML Tool	Dataset	Performance Measures
DDoS	[56]	Packet traffic	Neural network	Vanderbilt Prowler SD	Mean Square Error
	[57]	Access control	Multivariate correlation analysis	KDD Cup 99	Accuracy, FPR, TNR, Detection Rate
	[58]	Data traffic	SVM	SD using the probabilistic wireless network simulator (Prowler)	Accuracy, Time taken
	[59]	Data traffic	Neural network	NSL-KDD	Accuracy, Precision, Recall, F1 score
	[60]	Packet flooding	Random Forest	SD of TCP and UDP flood attacks	Precision, F1 score, Accuracy, Error rate, Mattews correlation coefficient
	[61]	Malicious traffic and volume	SVM	KDD99	Detection Rate Mean Threshold
	[62]	Malicious data traffic	Extensive Learning Machine (ELM) using neural network	CTU dataset	Accuracy, Precision, Recall, F1 score
Eavesdropping	[63]	Authentication	Non-parametric Bayesian	RSSI data	Proximity passing rate, number of breaches correctly identified by the algorithm proposed
Malware	[64]	Access control	KNN	Public (MalGenome) and Private (Self-collected)	Precision, Recall, F-measure, TPR, FPR
	[65]	Resource information	SVM	Real time data from normal and malicious applications	Accuracy, Precision, FPR
	[66]	Access control	Ensemble learning	Android malware samples using Java based APK analysis tool	Accuracy FPR, FNR, TPR, TNR, Precision
Intrusion	[67]–[69]	Access control	SVM	TCPDUMP and BSM SD [67], DARPA dataset from the Lincoln Laboratory of MIT [68], Real time dataset obtained from ISP [69]	[Attack Detection Rate, FPR [67]], [TPR, FPR, Precision, Recall, Accuracy [68]], [Accuracy, FPR, TNR [69]]
	[70]–[75]	Traffic anomaly	K-NN [70]–[72], [K-NN + Particle Swarm Optimization [73]], [Generic Algorithm + KNN [74]]	1998 DARPA BSM audit data [70], KDD Cup 1999 dataset [71], KDD-CUP 1999 dataset [72]–[74]	[FPR, Attack Detection Rate [70]], Accuracy [71], Accuracy [72], [73], [TPR, TNR, FPR, FNR, Accuracy [74]]
	[76], [77]	Traffic anomaly	Decision Tree [76], [Hybrid model with Decision Tree + SVM [77]]	KDDCup99 dataset [76], NSL-KDD data set [77]	[Accuracy, Precision, FPR, FNR, TPR, TNR [76]], [FPR, detection rate [77]]
	[108]-[109] [78], [79]	Anomalous packets	Naive Bayes	KDD'99 [78], NSL KDD dataset [79]	[Detection rate, FPR [78]], [Accuracy, Root Mean Squared Error (RMSE), TPR [79]]

S. Singh, et. Al. "Machine Learning Assisted Security and Privacy Provisioning for Edge Computing: A Survey", 2021.

Working Groups on the IoT Security

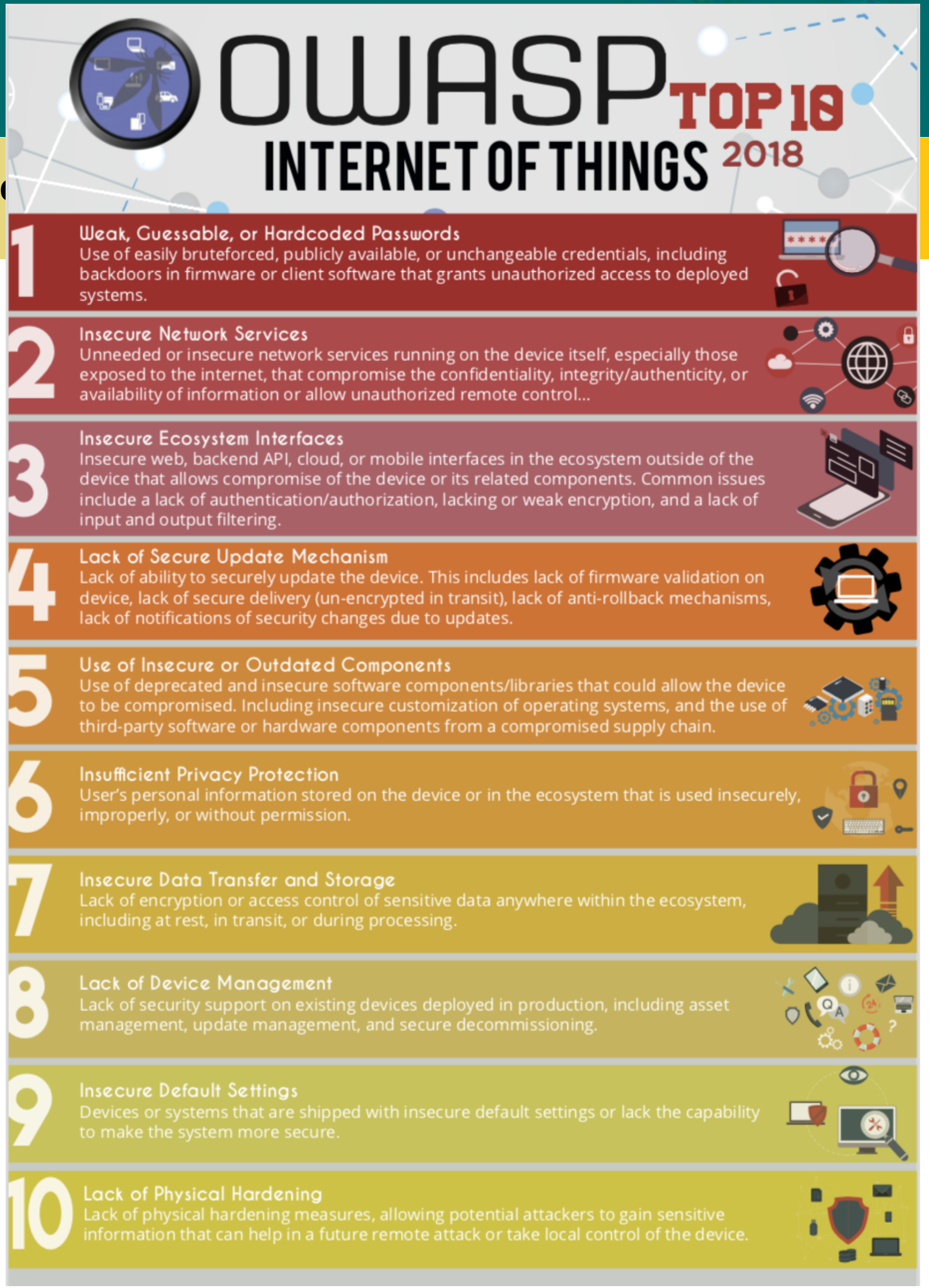
- OWASP IoT Security Guidance (https://owasp.org/www-project-internet-of-things/)
  - IAB Privacy & Security studies
  - RFC 7452 – Architectural Considerations in Smart Object Networking
  - RFC 7456 – Cryptographic algorithm agility
- EU Article 29 Data Protection Working party
  - Anonymization, privacy and the IoT
- NIST (National Institute of Standards and Technology) - Cyber Security Framework
- Track the emerging standards, e.g.
  - W3C Security Activity
  - IETF ACE & JOSE

OWASP IoT Top 10 - 2018

https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=I

OWASP IoT Top 10 2014	OWASP IoT Top 10 2018 Mapping
I1 Insecure Web Interface	I3 Insecure Ecosystem Interfaces
I2 Insufficient Authentication/Authorization	I1 Weak, Guessable, or Hardcoded Passwords I3 Insecure Ecosystem Interfaces I9 Insecure Default Settings
I3 Insecure Network Services	I2 Insecure Network Services
I4 Lack of Transport Encryption/Integrity Verification	I7 Insecure Data Transfer and Storage
I5 Privacy Concerns	I6 Insufficient Privacy Protection
I6 Insecure Cloud Interface	I3 Insecure Ecosystem Interfaces
I7 Insecure Mobile Interface	I3 Insecure Ecosystem Interfaces
I8 Insufficient Security Configurability	I9 Insecure Default Settings
I9 Insecure Software/Firmware	I4 Lack of Secure Update Mechanism I5 Use of Insecure or Outdated Components
I10 Poor Physical Security	I10 Lack of Physical Hardening

Ref: https://scriptingxss.gitbook.io/owasp-iot-top-10-mapping-project/

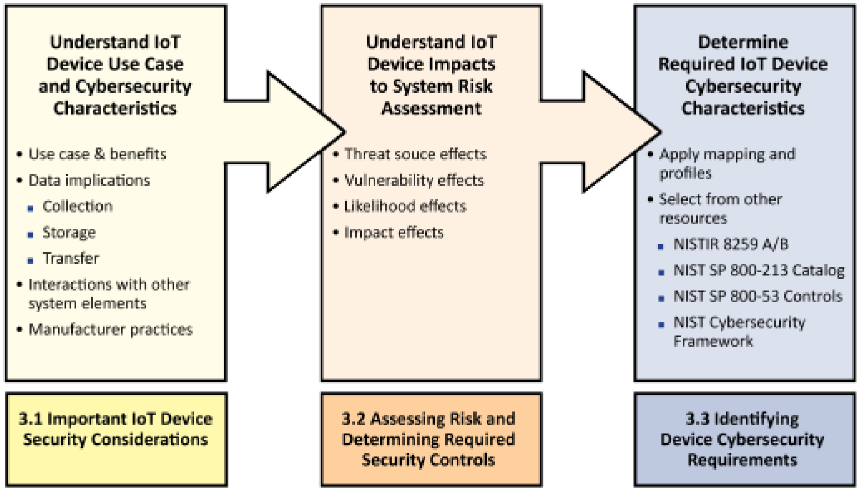


OWASP IoT Top 10 - 2018

NIST CSF

NIST: IoT Cybersecurity Framework

- NIST Cyber Security Framework (CSF): Identify-Protect-Detect-Respond-Recover
- NIST Computer Security Resource Center: https://csrc.nist.gov/
  - NIST Cybersecurity for IoT Program: https://www.nist.gov/itl/applied-cybersecurity/nist-cybersecurity-iot-program
  - Securing the Industrial Internet of Things: Cybersecurity for Distributed Energy Resources: https://www.nccoe.nist.gov/sites/default/files/2021-09/es-iiot-nist-sp1800-32-draft.pdf, September 2021.
  - Methodology for Characterizing Network Behavior of Internet of Things Devices: https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8349-draft.pdf
  - NIST Special Publication (SP) 800-213: **IoT Device Cybersecurity Guidance** for the Federal Government: Establishing IoT Device Cybersecurity Requirements; https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-213.pdf, November 2021.



NIST SP.800.213

The IoT Security Recommendations

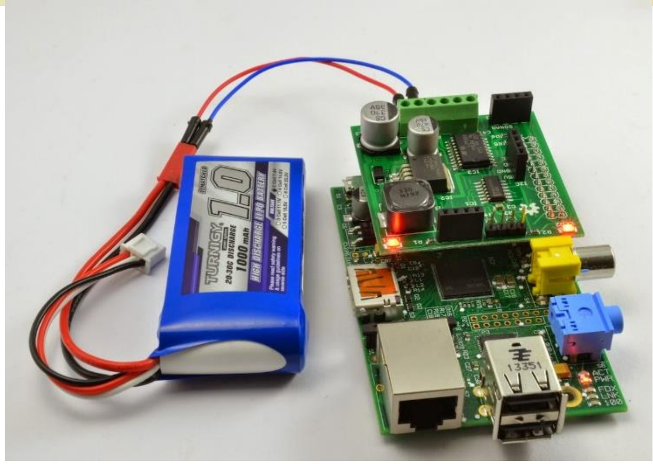
- Accommodate IoT with existing practices:
  - Policies, Procedures, & Standards
  - Awareness Training
  - Risk Management
  - Vulnerability Management
  - Forensics
- Plan for IoT growth:
  - Additional types of logging, log storage: Can you find the needle in the haystack?
  - Increased network traffic: will your firewall / IDS / IPS be compatible and keep up?
  - Increased demand for IP addresses both IPv4 and IPv6
  - Increased network complexity – should these devices be isolated or segmented?

References

- H. F. Atlam and G. B. Wills, “IoT Security, Privacy, Safety, and Ethics”, in Digital Twin Technologies and Smart Cities, Springer, 2020.
- Presentation Slides: “CSG Business Meeting:
  End-to-End Trust & Security Open Architecture for IoT” by Scot Ransbottom, Virginia Tech and Florence Hudson, Internet2, April 28, 2016.
- Tackling Data Security and Privacy Challenges for the Internet of Things, Dave Raggett, June 14, 2016.
- Brian Russell and Drew Van Duren, Practical Internet of Things Security, Packt Publishing, 2017
- Shancang Li and Li Da Xu, Securing the Internet of Things, Packt Publishing, 2017
- Maurice Dawson, Mohamed Eltayeb and Marwan Omar, Security Solutions for Hyperconnectivity and the Internet of Things, IGI Global, 2017.
- N. Jeyanthi and R. Thandeeswaran, Security Breaches and Threat Prevention in the Internet of Things, IGI Global, 2017.
- https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project
- https://www.experfy.com/training/courses/securing-enterprise-internet-of-things-implementations
- https://www.iotsecurityfoundation.org/
- https://users.ece.cmu.edu/~vsekar/Teaching/Fall21/18739//lectures/
- https://www.coursera.org/lecture/iot/lecture-3-2-risks-privacy-and-security-GnJON
- https://www.iot-now.com/2021/11/24/115797-arrow-electronics-psa-certified-development-kit-accelerates-time-to-market-for-iot-devices/?fbclid=IwAR1ir_fRs-D4gvbWujhT4_04PCOjdIrfrz8mhSmbSg4qm-WkB-ZWnJTNFhA

Part 3
Battery Life & Power Management

Example 1


- Battery Life (Battery Last) – (hr)=

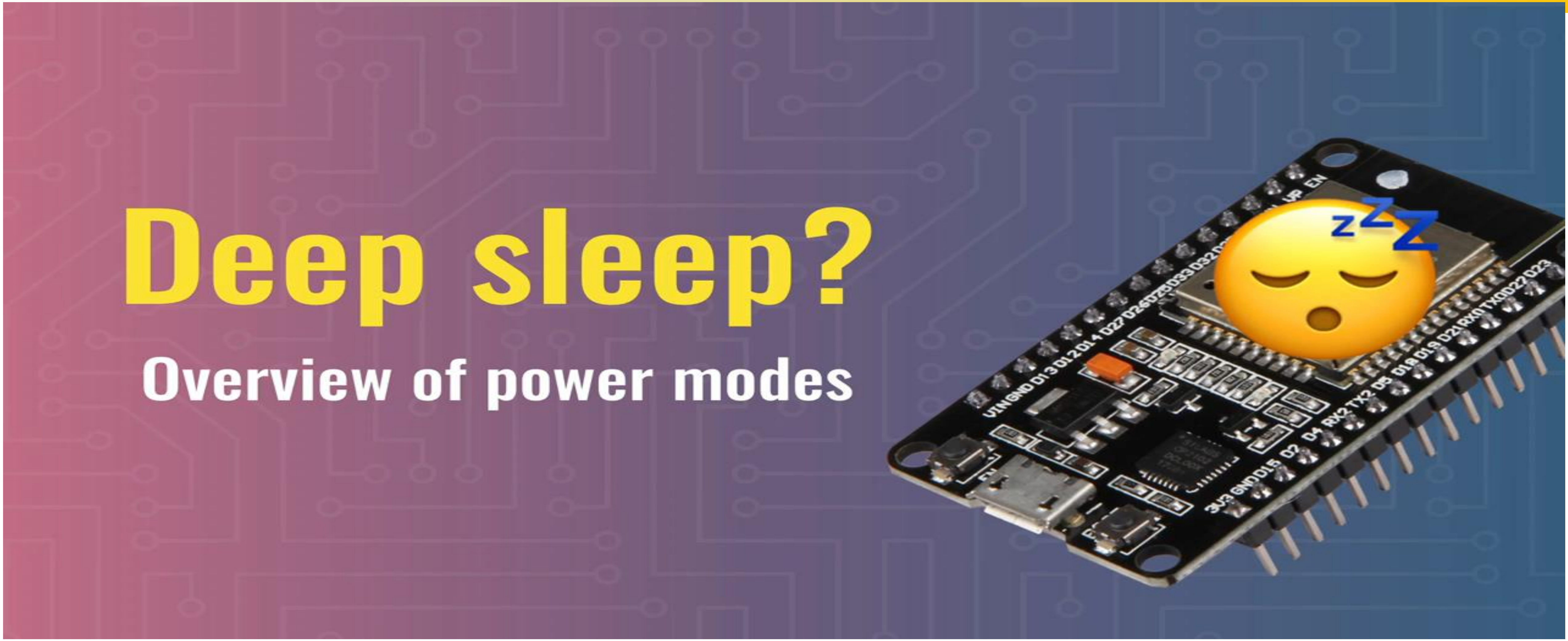
  Battery Capacity in mAh/ Load(Draw,consumption) Current in mA

- If are pessimistic and assume that the Pi will use 800mA (with WiFi + a couple of peripherals+CPU processing programs),

  then we can expect about 1000 / 800 = 1.25 hours of operation from a 1000mA battery.

Ref: https://electronics977.rssing.com/chan-8359630/latest.php

Power Modes



ESP32 Power Management

ESP32

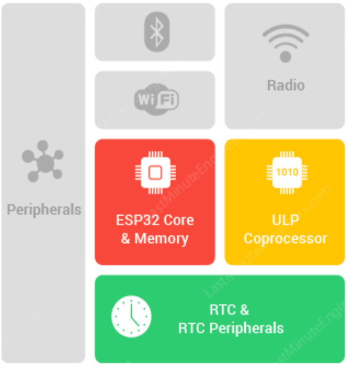
Active Mode

Active:
- WiFi
- Bluetooth
- Radio
- ESP32 Core
- ULP Coprocessor
- Peripherals
- RTC

Inactive:

Power Consumption:
160~260mA

Modem Sleep

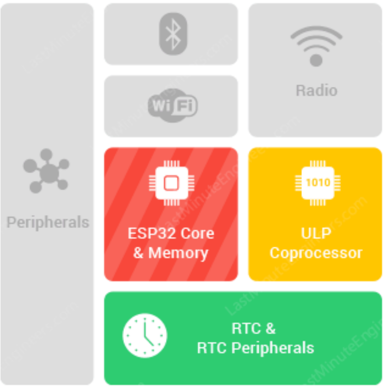


Active:
- ESP32 Core
- ULP Coprocessor
- RTC

Inactive:
- WiFi
- Bluetooth
- Radio
- Peripherals

Power Consumption:
3~20mA

Light Sleep



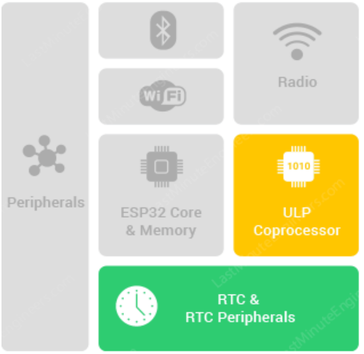
Active:
- ULP Coprocessor
- RTC

Paused:
- ESP32 Core

Inactive:
- WiFi
- Bluetooth
- Radio
- Peripherals

Power Consumption:
0.8mA

Deep Sleeep

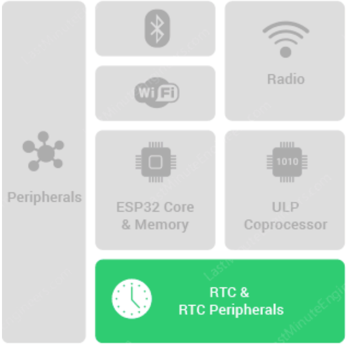


Active:
- ULP Coprocessor
- RTC

Inactive:
- ESP32 Core
- WiFi
- Bluetooth
- Radio
- Peripherals

Power Consumption:
10μA

Hibernation



Active:
- RTC

Inactive:
- ESP32 Core
- ULP Coprocessor
- WiFi
- Bluetooth
- Radio
- Peripherals

Power Consumption:
2.5μA

https://lastminuteengineers.com/esp32-sleep-modes-power-consumption/
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/power_management.html

Battery Life Computation

Example 2

Based on ESP32 Ultra-Low-Power (ULP) Management

Suppose, for the deep-sleep operation, 175 µA * 24 h = 4.2 mAh per day,
For the transmission bursts 140 mA * 1 s / 3600 s / h * 24 = 0.93 mAh,
Therefore, the total battery consumption 5.13 mAh per day.

Based on the battery capacity (1350mA),
this gives a calculated service life of 263.16 days.

http://www.of-things.de/battery-life-calculator.php
https://www.macnica.eu/
https://diyi0t.com/reduce-the-esp32-power-consumption/
https://community.hiveeyes.org/t/low-power-esp32-hardware-and-software/538