



life.augmented

A night-time aerial view of a city skyline with numerous skyscrapers. Overlaid on the image is a network diagram with white nodes and lines connecting them, and several blue Wi-Fi signal icons, representing an IoT network.

STM32U5 AWS IoT Hands-on

Slim JALLOULI

October 2022

Agenda

1 Clone the workshop repo

2 STM32U5 Introduction

3 FreeRTOS STM32U5 Reference Integration

4 STM32U5 AWS QuickConnect

5 Boards distribution

6 Hands-on

1- Clone the Workshop Repo

Clone the workshop repo

- You can access the repo using this url: <https://tinyurl.com/stm32u5aws>
- Clone the workshop repo:

```
git clone https://github.com/SlimJallouli/ST_DevCon_2022_AWS_Workshop.git --recurse-submodules
```

2- STM32U5 Introduction

STM32U5 Microcontrollers

The new reference for secure and smart IoT applications



Higher Security

Certified PSA L3 and SESIP L3

Lower Power

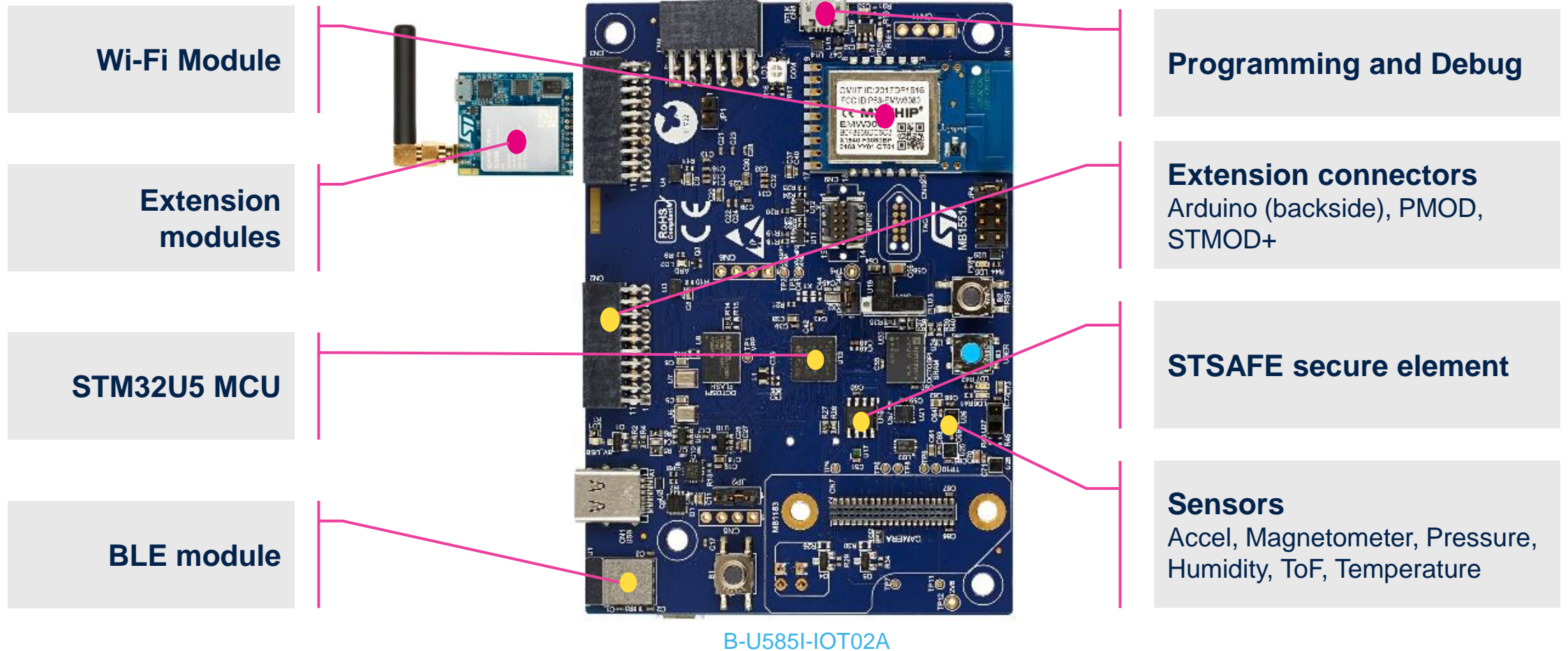
58 ULPMark-CM

Richer applications

Cortex-M33 @ 160MHz, extended features set

STM32U5 IoT Kit

Your reference board for IoT Proof-of-Concept



AWS IoT on STM32U5

X-CUBE-AWS reference integration simplifying your development



STM32
CubeExpansion
X-CUBE-AWS

AWS Certified

Leveraging ARM Trusted Firmware-M (TF-M)

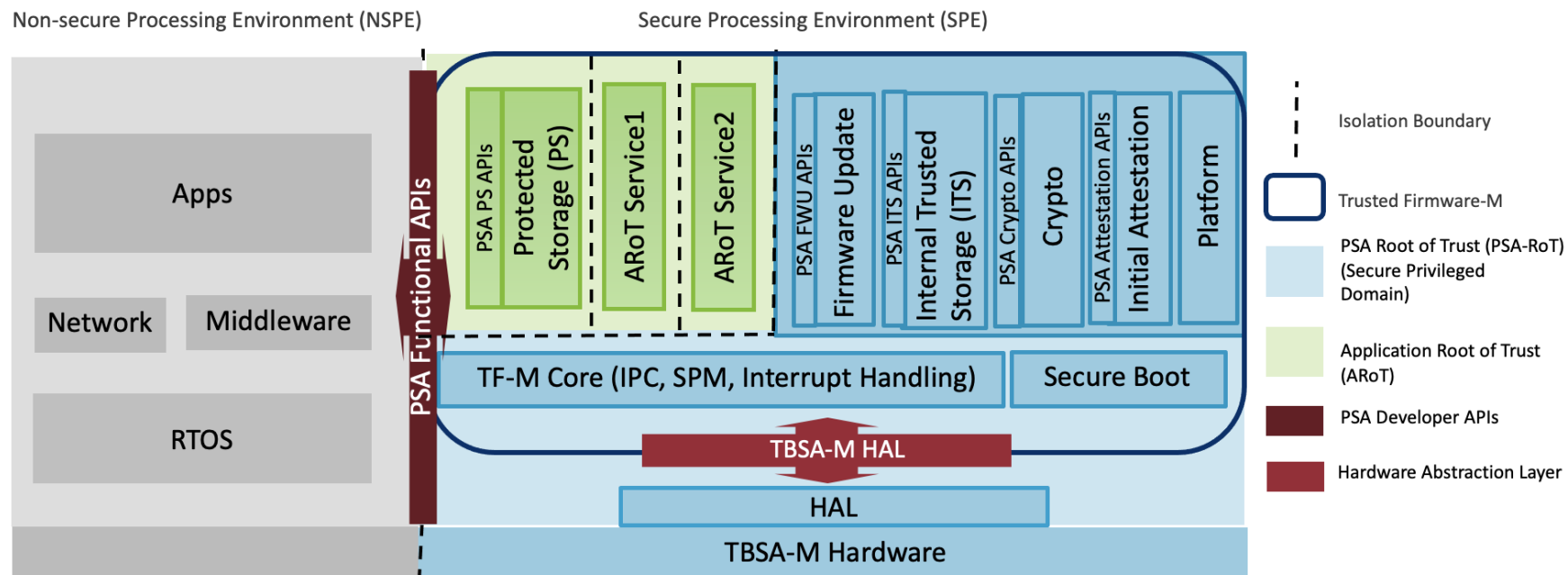
Based on FreeRTOS LTS Library

Over The Air Update

AWS IoT Defender

Trusted Firmware-M (TF-M)

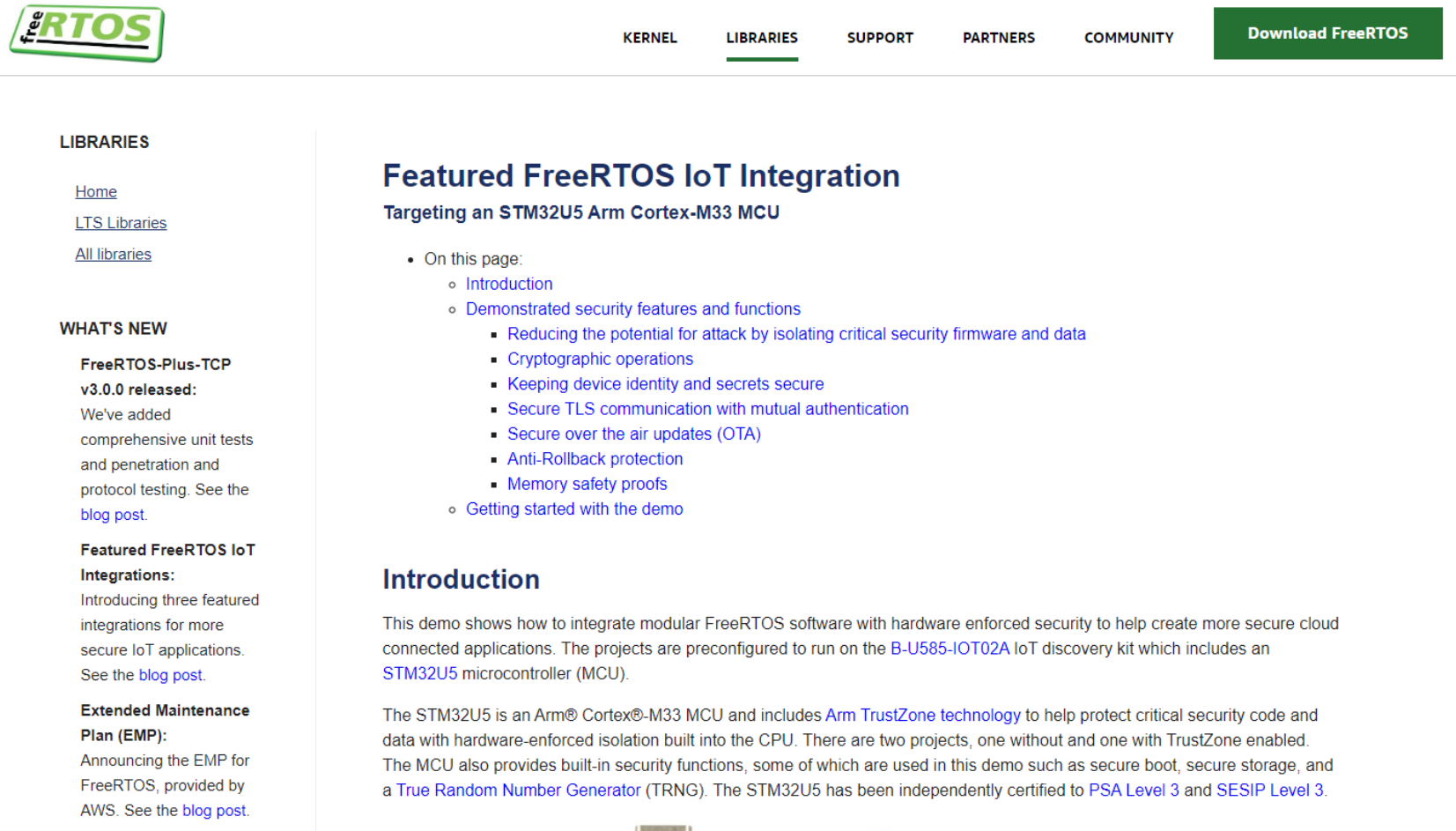
- Trusted Firmware for Cortex M (TF-M) implements the Secure Processing Environment (SPE) for Armv8-M, Armv8.1-M
- **Trusted Firmware-M consists of:**
 - Secure Boot, Control the isolation, communication and execution within SPE and with NSPE
 - Crypto, Internal Trusted Storage (ITS), Protected Storage (PS) and Attestation secure services



3- STM32U5 FreeRTOS reference

Featured FreeRTOS IoT Integration

- <https://www.freertos.org/STM32U5/>



The screenshot shows the FreeRTOS website with the 'LIBRARIES' tab selected in the navigation bar. The main content area is titled 'Featured FreeRTOS IoT Integration' and 'Targeting an STM32U5 Arm Cortex-M33 MCU'. It includes a list of features and an introduction to the demo.

LIBRARIES

- [Home](#)
- [LTS Libraries](#)
- [All libraries](#)

WHAT'S NEW

FreeRTOS-Plus-TCP v3.0.0 released:
We've added comprehensive unit tests and penetration and protocol testing. See the [blog post](#).

Featured FreeRTOS IoT Integrations:
Introducing three featured integrations for more secure IoT applications. See the [blog post](#).

Extended Maintenance Plan (EMP):
Announcing the EMP for FreeRTOS, provided by AWS. See the [blog post](#).

Featured FreeRTOS IoT Integration
Targeting an STM32U5 Arm Cortex-M33 MCU

- On this page:
 - [Introduction](#)
 - [Demonstrated security features and functions](#)
 - Reducing the potential for attack by isolating critical security firmware and data
 - Cryptographic operations
 - Keeping device identity and secrets secure
 - Secure TLS communication with mutual authentication
 - Secure over the air updates (OTA)
 - Anti-Rollback protection
 - Memory safety proofs
 - [Getting started with the demo](#)

Introduction

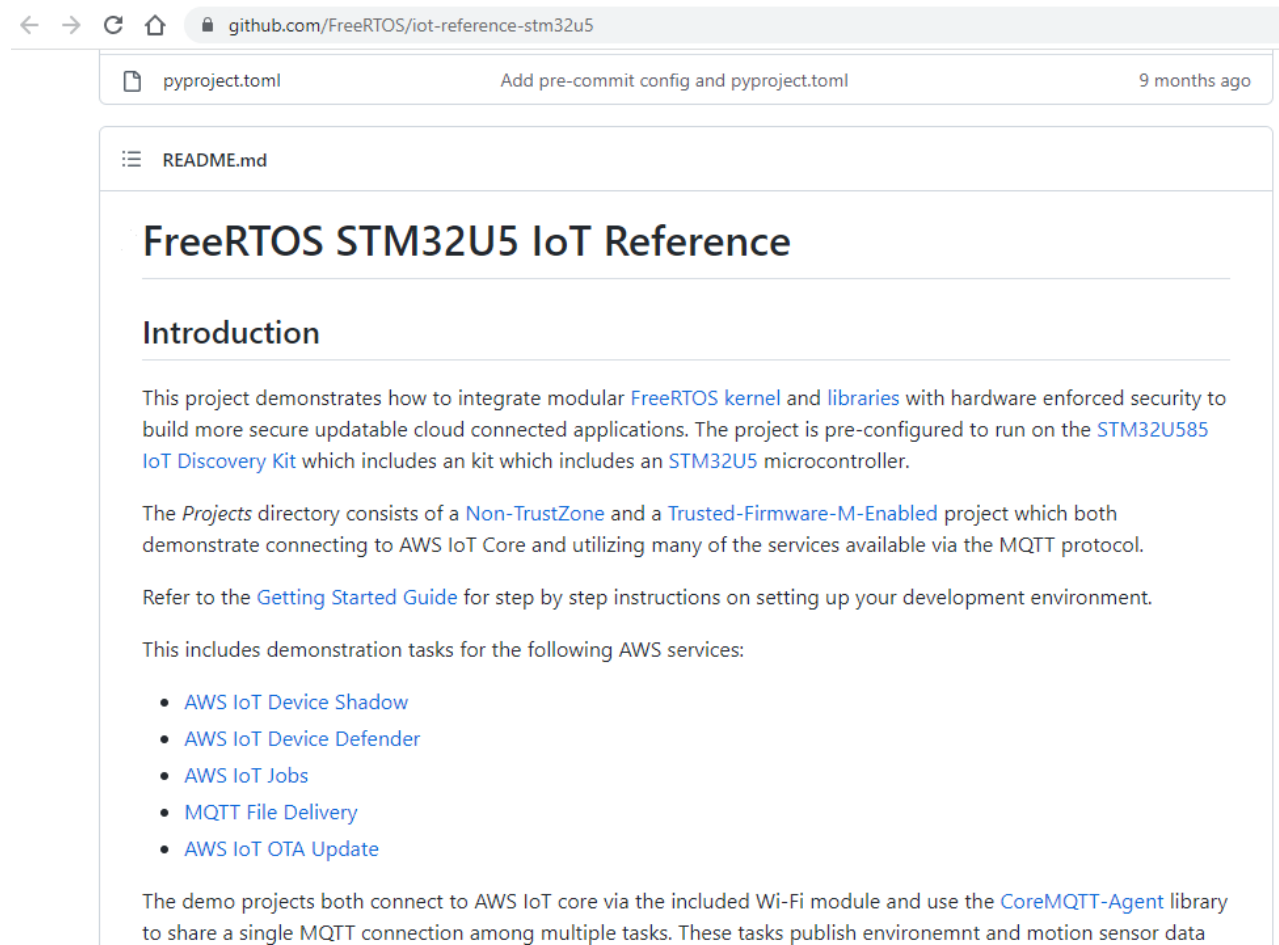
This demo shows how to integrate modular FreeRTOS software with hardware enforced security to help create more secure cloud connected applications. The projects are preconfigured to run on the [B-U585-IOT02A](#) IoT discovery kit which includes an [STM32U5](#) microcontroller (MCU).

The STM32U5 is an Arm® Cortex®-M33 MCU and includes [Arm TrustZone technology](#) to help protect critical security code and data with hardware-enforced isolation built into the CPU. There are two projects, one without and one with TrustZone enabled. The MCU also provides built-in security functions, some of which are used in this demo such as secure boot, secure storage, and a [True Random Number Generator](#) (TRNG). The STM32U5 has been independently certified to [PSA Level 3](#) and [SESIP Level 3](#).

FreeRTOS STM32U5 GitHub repository

- <https://github.com/FreeRTOS/iot-reference-stm32u5>

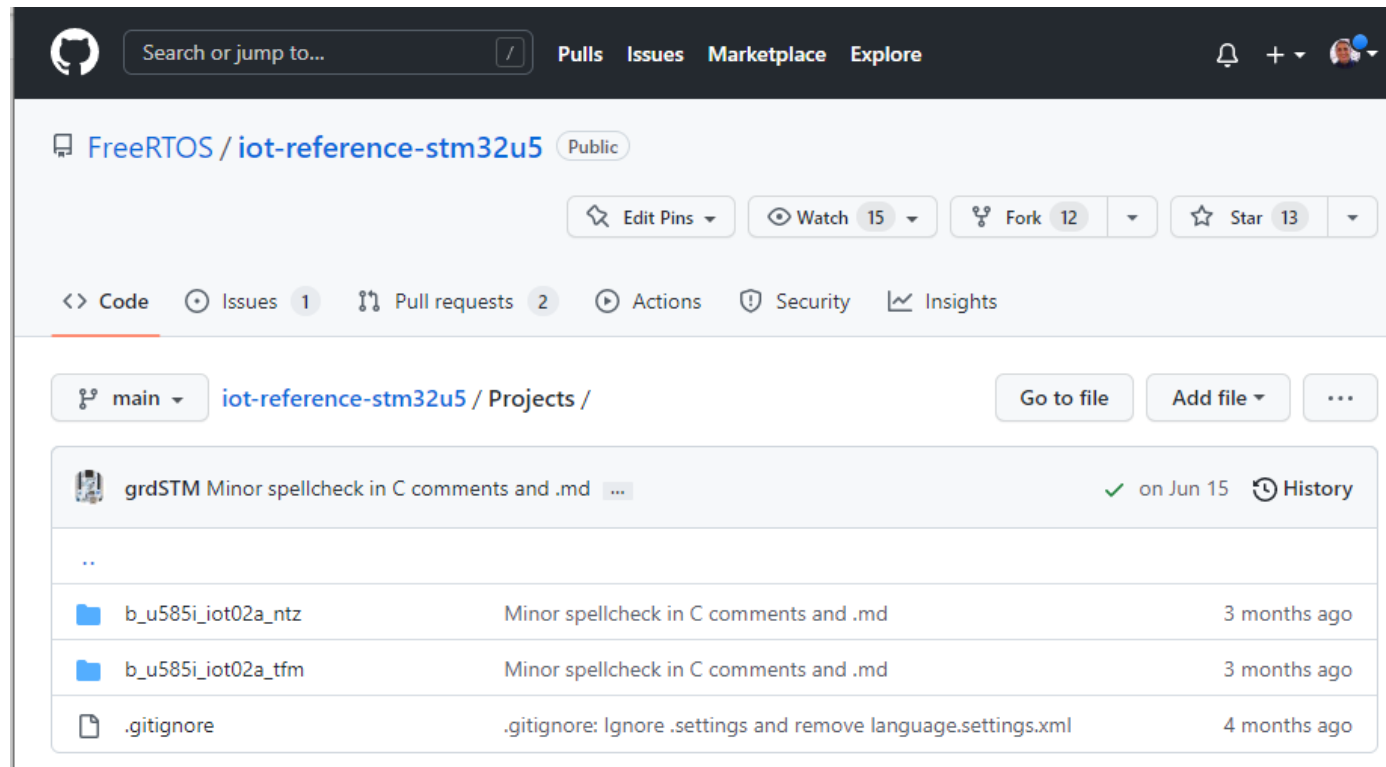
```
git clone https://github.com/FreeRTOS/iot-reference-stm32u5.git --recurse-submodules
```



The screenshot shows the GitHub repository page for `FreeRTOS/iot-reference-stm32u5`. The browser address bar displays `github.com/FreeRTOS/iot-reference-stm32u5`. Below the header, there is a commit history bar showing a commit for `pyproject.toml` titled "Add pre-commit config and pyproject.toml" from 9 months ago. The main content area shows the `README.md` file. The README title is "FreeRTOS STM32U5 IoT Reference". Under the "Introduction" section, it describes the project's goal: to integrate the FreeRTOS kernel and libraries with hardware-enforced security for secure, updatable cloud-connected applications on the STM32U585 IoT Discovery Kit. It mentions that the `Projects` directory contains a `Non-TrustZone` and a `Trusted-Firmware-M-Enabled` project, both demonstrating connections to AWS IoT Core via MQTT. It refers to a `Getting Started Guide` for setup instructions and lists demonstration tasks for AWS services: AWS IoT Device Shadow, AWS IoT Device Defender, AWS IoT Jobs, MQTT File Delivery, and AWS IoT OTA Update. The bottom of the README states that demo projects connect to AWS IoT Core via Wi-Fi and use the `CoreMQTT-Agent` library to share a single MQTT connection among multiple tasks, publishing environment and motion sensor data.

Projects

- Two projects
 - b_u585i_iot02a_ntz (For POC only)
 - b_u585i_iot02a_tfm (For production)

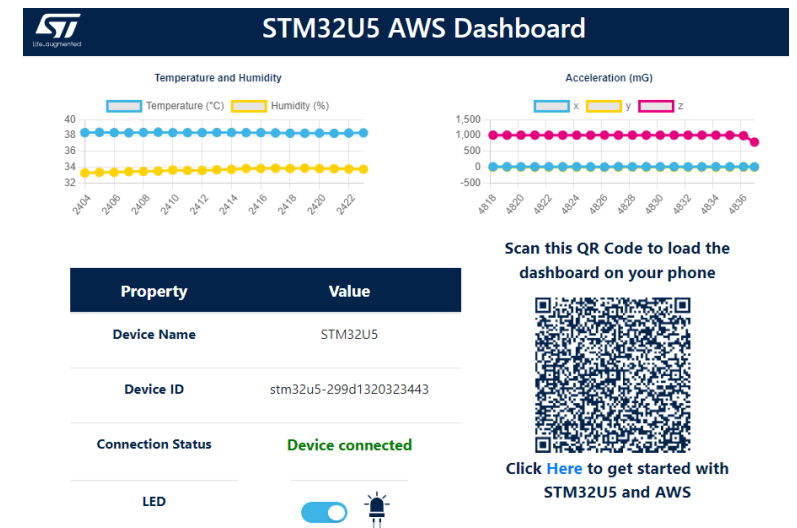


4- STM32U5 AWS Quick Connect

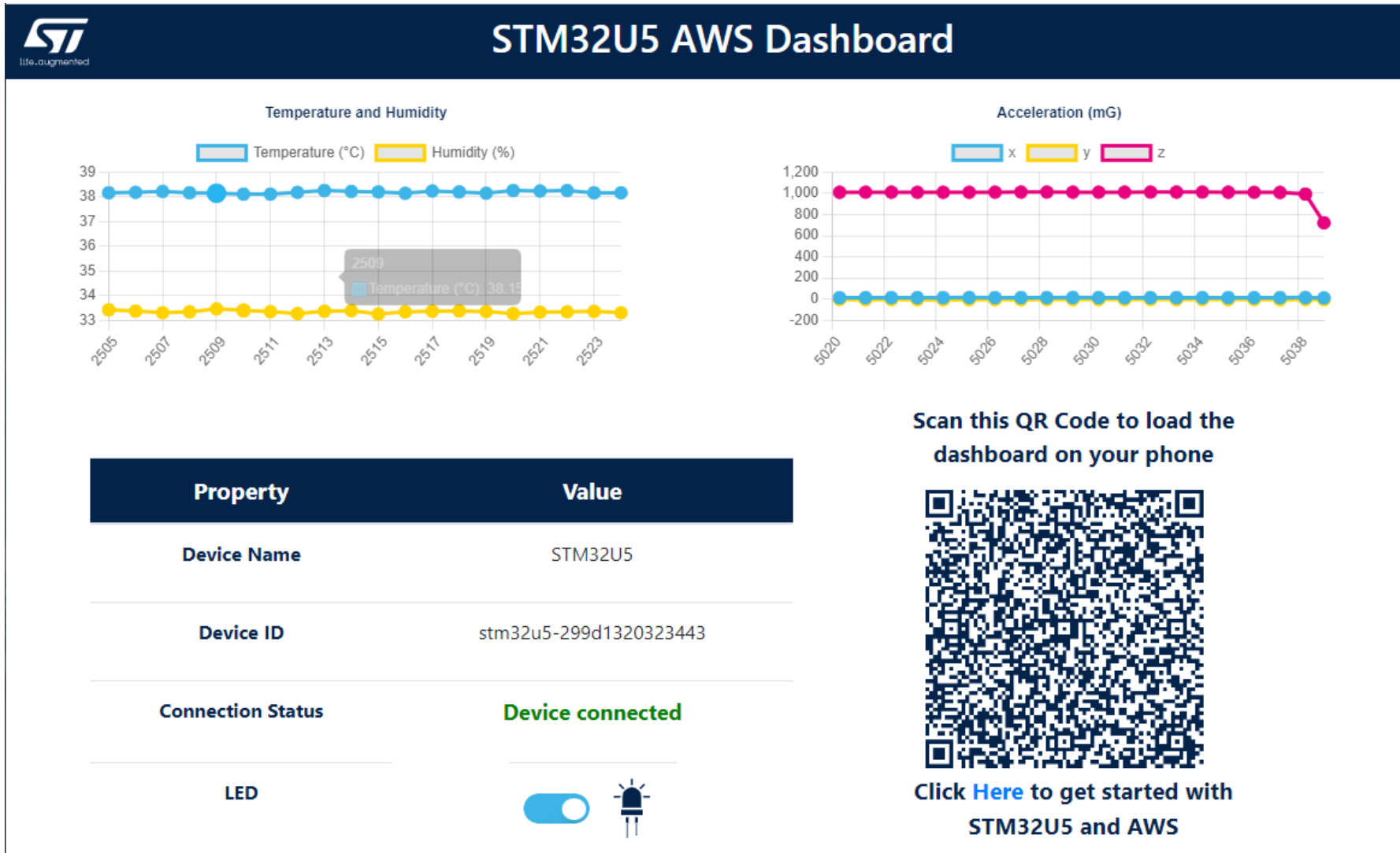
AWS Quick Connect

Abstracts Firmware customization and registration process

- Allows Cloud IoT/Data architects to focus on developing power of the Cloud IoT platform proof of concepts.
- Connect to AWS IoT and perform telemetry in minutes
- Solution Components:
 - *B-U585I-IOT02A Discovery Kit*
 - *Reference Binary*
 - *Quick connect scripts*
 - *Cloud visualization*



STM32U AWS Dashboard



5- Boards distribution

Boards distribution

- Believe it or Not, even at ST we have hard time to get boards. Unfortunately, we need to collect the board at the end of the workshop to use them in the next one.
- We'll disinfect the boards at the end of the workshop and before re-distribution.
- A voucher is handed at the end of the workshop



6- Labs

Lab 1: System preparation

Lab 1: System preparation

- In this lab we'll make sure that all the tools are properly installed and that your PC is ready to run the STM32U5_AWS_QuickConnect script

System Check

- Navigate to `ST_DevCon_2022_AWS_Workshop`
- Open a command window (example PowerShell or bash)

- Run:

- `aws --version`
- `python --version`

Windows PowerShell

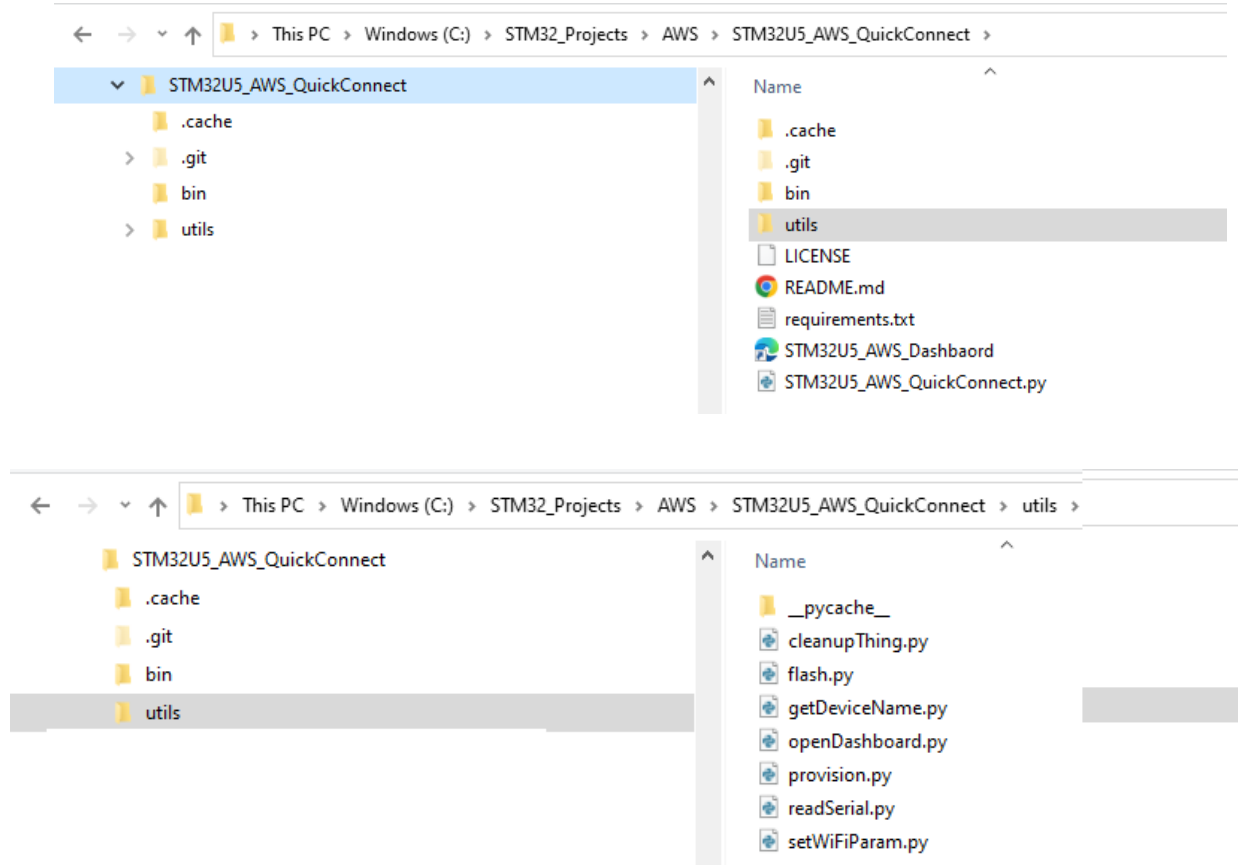
```
PS C:\STM32_Projects\AWS\STM32U5_AWS_QuickConnect> aws --version
aws-cli/2.0.53 Python/3.7.7 windows/10 exe/AMD64
PS C:\STM32_Projects\AWS\STM32U5_AWS_QuickConnect> python --version
Python 3.10.4
PS C:\STM32_Projects\AWS\STM32U5_AWS_QuickConnect> _
```

- Navigate to the `STM32U5_AWS_QuickConnect` directory
- Run: `pip install -r requirements.txt`

AWS CLI profiles

- Navigate to the `STM32_AWS_QuickConnect` directory
- Open a command window (example PowerShell or bash)
 - For Windows users double click on `AWS_CLI_ProfileConfig.bat`
 - For Linux and MAC users run `AWS_CLI_ProfileConfig.sh`
- The scripts will add two AWS CLI profiles called `provision` and `dashboard`.
- The first profile is used to provision your board with AWS IoT core
- The second profile is used to open the STM32U5 AWS Dashboard

STM32U5 AWS QuickConnect



`pip install -r requirements.txt`

Link to your device dashboard

STM32U5 QuickConnect script

Utils:

- Flash the binary
- Generate a device name
- Change Wi-Fi ssid and password
- Provision the board
- Open dashboard and create shortcut
- Read and print the serial port

Lab 2: Connect to AWS IoT Core

Lab 2: Connect to AWS IoT Core

- In this lab we'll use the STM32_AWS_QuickConnect to connect your board AWS IoT Core and open a dashboard to visualize the sensor data and control the LED.

Connect your board

- Connect your board to the PC



Run the quick connect script

- Navigate to `STM32U5_AWS_QuickConnect` directory
- Open a PowerShell console
- Type `python .\STM32U5_AWS_QuickConnect.py -i`
- Accept all the default settings
- The script will:
 - Flash your board with the binary
 - Provision your board with AWS IoT Core
 - Set the Wi-Fi SSID and password
 - Create a shortcut link to the dashboard specifically for your board
 - Open the dashboard for your board

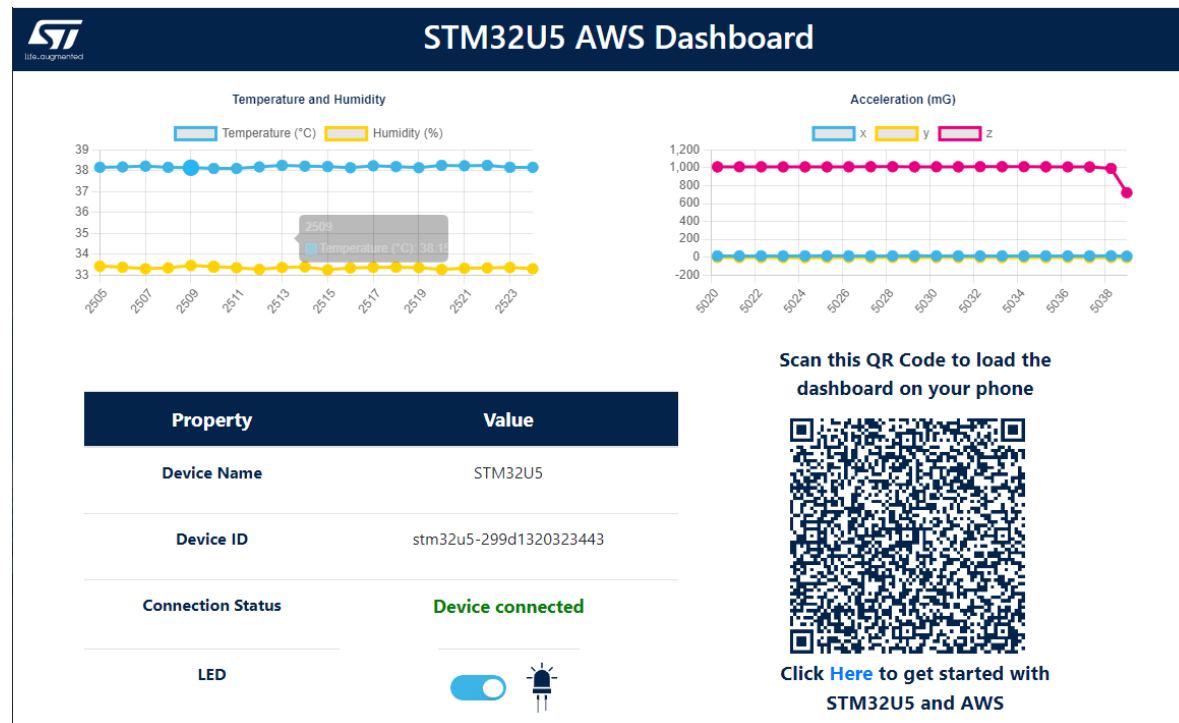
Main Dashboard

- I'll have a special dashboard showing the number of connected devices, the device ID and the corresponding LED status in real time as your boards get connected.



STM32U5 AWS Dashboard

- Scan the QR code with your phone camera
- Move the board to see the sensor data changing
- Use the toggle button to toggle the LED On/Off



Lab 3: FreeRTOS-Plus-CLI

Lab 3: FreeRTOS-Plus-CLI

- In this lab we'll use the **FreeRTOS-Plus-CLI** to check and change the board configuration and check the application status

FreeRTOS-Plus-CLI

https://www.freertos.org/FreeRTOS-Plus/FreeRTOS_Plus_CLI/FreeRTOS_Plus_Command_Line_Interface.html

The screenshot shows the FreeRTOS-Plus-CLI website. The top navigation bar includes links for KERNEL, LIBRARIES (which is highlighted), SUPPORT, PARTNERS, and COMMUNITY, along with a 'Download FreeRTOS' button. The left sidebar contains a 'LIBRARIES' section with links to Home, Getting started, All libraries, and a sub-section for FreeRTOS-Plus-CLI containing links to Introduction, Documentation, and Demos. Below this is a 'WHAT'S NEW' section for FreeRTOS-Plus-TCP v3.0.0 released, mentioning unit tests and a blog post, and a 'Featured FreeRTOS IoT Integrations' section.

The main content area is titled 'FreeRTOS-Plus-CLI' and 'An Extensible Command Line Interface Framework'. It features an 'Introduction' section stating that the framework provides a simple, small, extensible, and RAM-efficient method for processing command line input. It includes a clickable diagram with four stages:

- Provide a function that implements the command behaviour
- Map the command to the function that implements its behaviour
- Register the command with FreeRTOS+CLI
- Run the command interpreter

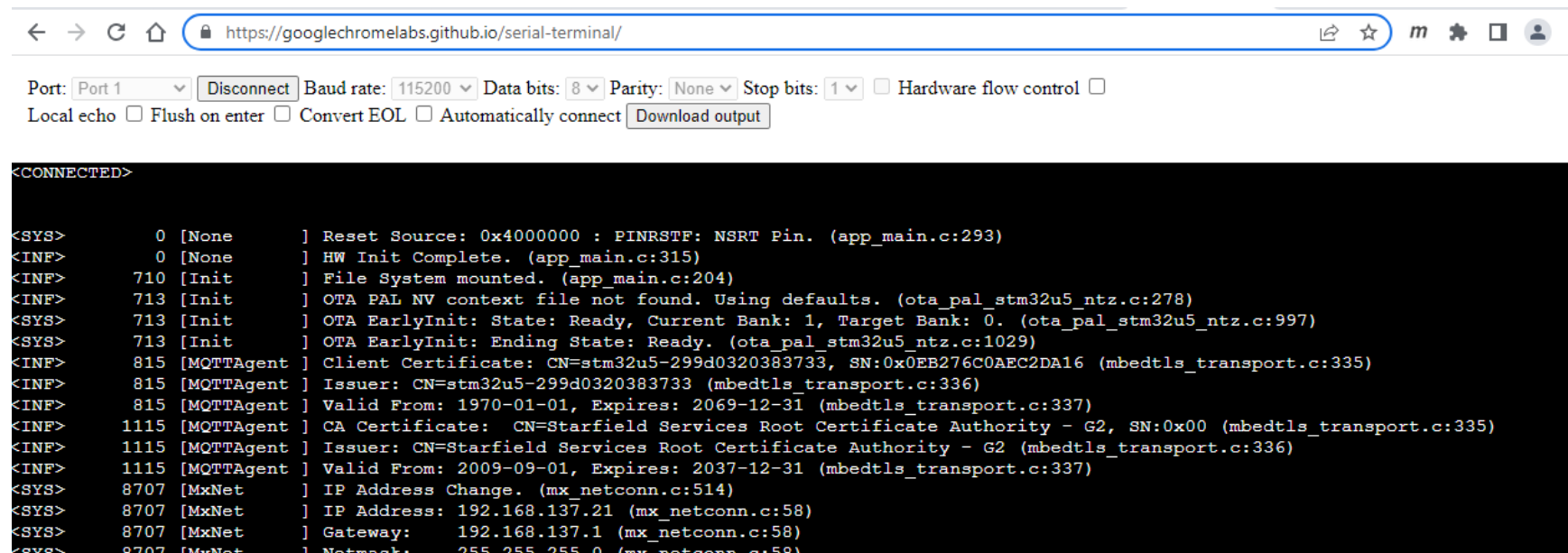
Below the diagram, it says: 'Adding a command to FreeRTOS-Plus-CLI. This diagram is clickable.'

Connect to the board over serial port

- Close the quick connect script window
- You can use a serial terminal like TeraTerm or this web based serial terminal

<https://googlechromelabs.github.io/serial-terminal/>

- Connect to the board (8-bits, 1-stop, 115200)



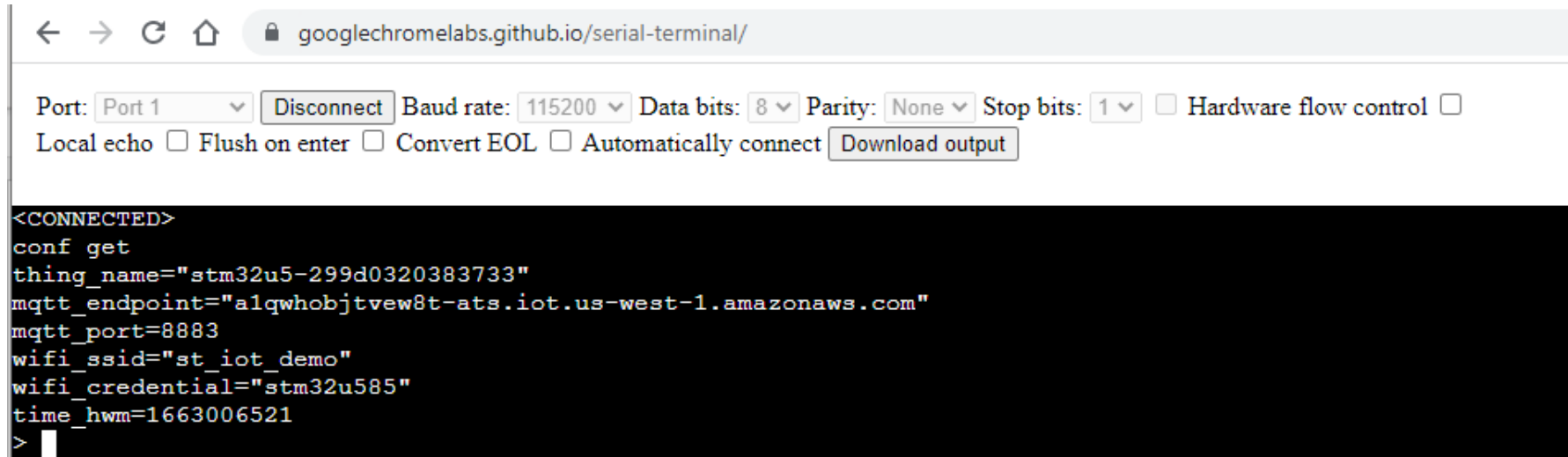
The screenshot shows a web browser window with the URL <https://googlechromelabs.github.io/serial-terminal/>. The interface includes a control bar with the following settings: Port: Port 1, Disconnect button, Baud rate: 115200, Data bits: 8, Parity: None, Stop bits: 1, Hardware flow control (unchecked), Local echo (unchecked), Flush on enter (unchecked), Convert EOL (unchecked), Automatically connect (unchecked), and a Download output button. Below the control bar is a terminal window displaying the following log output:

```
<CONNECTED>

<SYS>      0 [None      ] Reset Source: 0x4000000 : PINRSTF: NSRT Pin. (app_main.c:293)
<INF>      0 [None      ] HW Init Complete. (app_main.c:315)
<INF>     710 [Init       ] File System mounted. (app_main.c:204)
<INF>     713 [Init       ] OTA PAL NV context file not found. Using defaults. (ota_pal_stm32u5_ntz.c:278)
<SYS>     713 [Init       ] OTA EarlyInit: State: Ready, Current Bank: 1, Target Bank: 0. (ota_pal_stm32u5_ntz.c:997)
<SYS>     713 [Init       ] OTA EarlyInit: Ending State: Ready. (ota_pal_stm32u5_ntz.c:1029)
<INF>     815 [MQTTAgent ] Client Certificate: CN=stm32u5-299d0320383733, SN:0x0EB276C0AEC2DA16 (mbedtls_transport.c:335)
<INF>     815 [MQTTAgent ] Issuer: CN=stm32u5-299d0320383733 (mbedtls_transport.c:336)
<INF>     815 [MQTTAgent ] Valid From: 1970-01-01, Expires: 2069-12-31 (mbedtls_transport.c:337)
<INF>    1115 [MQTTAgent ] CA Certificate: CN=Starfield Services Root Certificate Authority - G2, SN:0x00 (mbedtls_transport.c:335)
<INF>    1115 [MQTTAgent ] Issuer: CN=Starfield Services Root Certificate Authority - G2 (mbedtls_transport.c:336)
<INF>    1115 [MQTTAgent ] Valid From: 2009-09-01, Expires: 2037-12-31 (mbedtls_transport.c:337)
<SYS>     8707 [MxNet      ] IP Address Change. (mx_netconn.c:514)
<SYS>     8707 [MxNet      ] IP Address: 192.168.137.21 (mx_netconn.c:58)
<SYS>     8707 [MxNet      ] Gateway: 192.168.137.1 (mx_netconn.c:58)
<SYS>     8707 [MxNet      ] Netmask: 255.255.255.0 (mx_netconn.c:58)
```

FreeRTOS CLI: Check your board configuration

- On the terminal type `conf get`



The screenshot shows a web browser window with the address bar displaying `googlechromelabs.github.io/serial-terminal/`. Below the address bar, there are configuration controls for a serial terminal: a dropdown menu for 'Port' (set to 'Port 1'), a 'Disconnect' button, a 'Baud rate' dropdown (set to '115200'), a 'Data bits' dropdown (set to '8'), a 'Parity' dropdown (set to 'None'), a 'Stop bits' dropdown (set to '1'), and a checkbox for 'Hardware flow control' (unchecked). Below these are checkboxes for 'Local echo' (unchecked), 'Flush on enter' (unchecked), 'Convert EOL' (unchecked), and 'Automatically connect' (unchecked), followed by a 'Download output' button. The main terminal area is a black rectangle with white text. It starts with `<CONNECTED>`, followed by the command `conf get` and its output: `thing_name="stm32u5-299d0320383733"`, `mqtt_endpoint="algwhobjtview8t-ats.iot.us-west-1.amazonaws.com"`, `mqtt_port=8883`, `wifi_ssid="st_iot_demo"`, `wifi_credential="stm32u585"`, and `time_hwm=1663006521`. The prompt `>` is visible at the bottom of the terminal area.

```
<CONNECTED>
conf get
thing_name="stm32u5-299d0320383733"
mqtt_endpoint="algwhobjtview8t-ats.iot.us-west-1.amazonaws.com"
mqtt_port=8883
wifi_ssid="st_iot_demo"
wifi_credential="stm32u585"
time_hwm=1663006521
>
```

FreeRTOS CLI: Change your board Wi-Fi settings

You can use the terminal and type the following commands

```
> conf set wifi_ssid myssid  
wifi_ssid="myssid"  
> conf set wifi_credential mypasswd  
wifi_credential="mypasswd"  
> conf commit  
Configuration saved to NVM.  
> reset
```

Use Help menu

- Type `help` for help menu and you will get the list of all possible command

```
> help
help:
  List available commands and their arguments.
  Usage:

  help
    Print help for all recognized commands

  help <command>
    Print help test for a specific command

conf:
  Get/ Set/ Commit runtime configuration values
  Usage:
  conf get
    Outputs the value of all runtime config options supported by the system.

  conf get <key>
    Outputs the current value of a given runtime config item.

  conf set <key> <value>
    Set the value of a given runtime config item. This change is staged
    in volatile memory until a commit operation occurs.

  conf commit
    Commit staged config changes to nonvolatile memory.

pki:
  Perform public/private key operations.
  Usage:
  pki <verb> <object> <args>
    Valid verbs are { generate, import, export, list }
    Valid object types are { key, csr, cert }
    Arguments should be specified in --<arg_name> <value>

  pki generate key <label_public> <label_private> <algorithm> <algorithm_param>
    Generates a new private key to be stored in the specified labels
```

Heap statistics

- Type `heapstat` to get info about the heap usage

```
> heapstat
+-----+
| Metric          | Dec (Bytes) | Hex (Bytes) | % Total |
+-----+-----+-----+-----+
| Heap Total      | 307200      | 0x4B000     | 100 %   |
| Heap Free       | 106960      | 0x1A1D0     | 34 %    |
| Min. Heap Free  | 93200       | 0x16C10     | 30 %    |
| Heap Alloc.     | 200240      | 0x30E30     | 65 %    |
| Max. Heap Alloc.| 214000      | 0x343F0     | 69 %    |
+-----+-----+-----+-----+
```

List running tasks and statistics

- Type `ps` to get info about the running tasks

```
> ps
Total Runtime: 81725
```

Task ID	State	Task Name	__Priority__		%CPU	Stack	Stack	Stack
			Base	Cur.		Alloc	HWM	Usage
4	RUNNING	cli	10	10	0%	2048	1906	6%
2	READY	IDLE	0	0	96%	1025	1001	2%
6	BLOCKED	uartTx	24	24	0%	1024	954	6%
15	BLOCKED	MotionS	5	5	0%	2048	1784	12%
5	BLOCKED	uartRx	30	30	0%	1024	990	3%
9	BLOCKED	lwIP	25	25	0%	4096	3964	3%
10	BLOCKED	MxData	25	25	0%	4096	4026	1%
14	BLOCKED	EnvSense	6	6	0%	1024	612	40%
7	BLOCKED	Heartbeat	0	0	0%	128	104	18%
8	BLOCKED	MxNet	23	23	0%	1024	838	18%
13	BLOCKED	OTAUpdate	1	1	0%	4096	3972	3%
17	BLOCKED	AWSDefender	5	5	0%	2048	1608	21%
12	BLOCKED	MQTTAgent	10	10	2%	2048	1394	31%
1	SUSPENDED	Init	8	8	0%	1024	738	27%
11	SUSPENDED	MxCtrl	24	24	0%	4096	3972	3%
19	BLOCKED	OTAAgent	3	3	0%	4096	3846	6%
16	BLOCKED	ShadowDevice	5	5	0%	1024	880	14%
3	BLOCKED	Tmr Svc	24	24	0%	2049	2019	1%

```
>
```

Reset (reboot) the system

- Type `reset` to reset the device

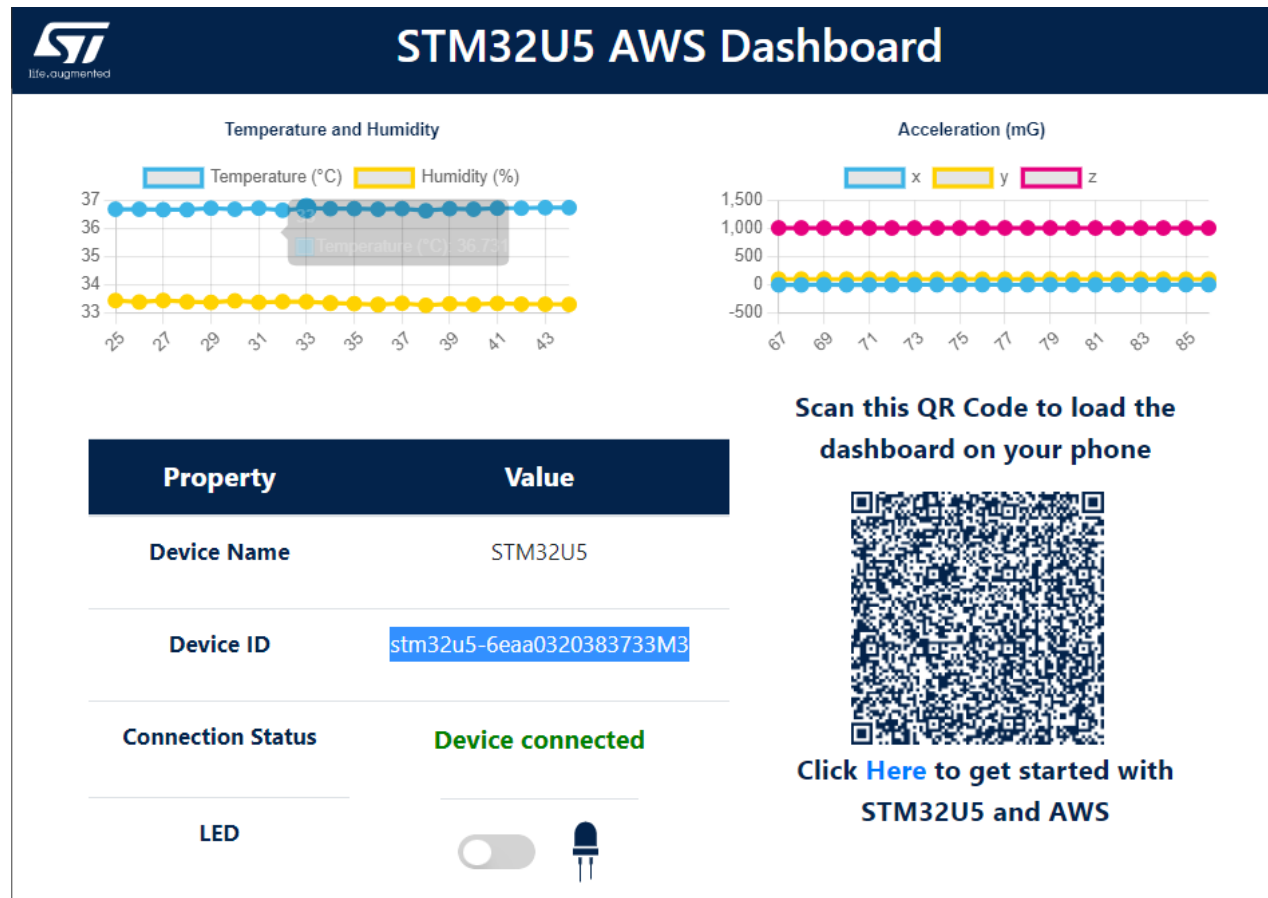
```
> reset
Resetting device.

<SYS>      0 [None      ] Reset Source: 0x14000000 : SFTRSTF: Software. (app_main.c:293)
<INF>      0 [None      ] HW Init Complete. (app_main.c:315)
<INF>     710 [Init      ] File System mounted. (app_main.c:204)
<INF>     713 [Init      ] OTA PAL NV context file not found. Using defaults. (ota_pal_stm32u5_ntz.c:278)
<SYS>     713 [Init      ] OTA EarlyInit: State: Ready, Current Bank: 1, Target Bank: 0. (ota_pal_stm32u5_ntz.c:997)
<SYS>     713 [Init      ] OTA EarlyInit: Ending State: Ready. (ota_pal_stm32u5_ntz.c:1029)
<INF>     809 [MQTTAgent ] Client Certificate: CN=stm32u5-6eaa0320383733M3, SN:0x00F23EE9F59262DF40 (mbedtls_transport.c:335)
<INF>     809 [MQTTAgent ] Issuer: CN=stm32u5-6eaa0320383733M3 (mbedtls_transport.c:336)
<INF>     809 [MQTTAgent ] Valid From: 1970-01-01, Expires: 2069-12-31 (mbedtls_transport.c:337)
<INF>    1109 [MQTTAgent ] CA Certificate: CN=Starfield Services Root Certificate Authority - G2, SN:0x00 (mbedtls_transport.c:335)
<INF>    1109 [MQTTAgent ] Issuer: CN=Starfield Services Root Certificate Authority - G2 (mbedtls_transport.c:336)
<INF>    1109 [MQTTAgent ] Valid From: 2009-09-01, Expires: 2037-12-31 (mbedtls_transport.c:337)
>
```


Lab 4: Device shadow

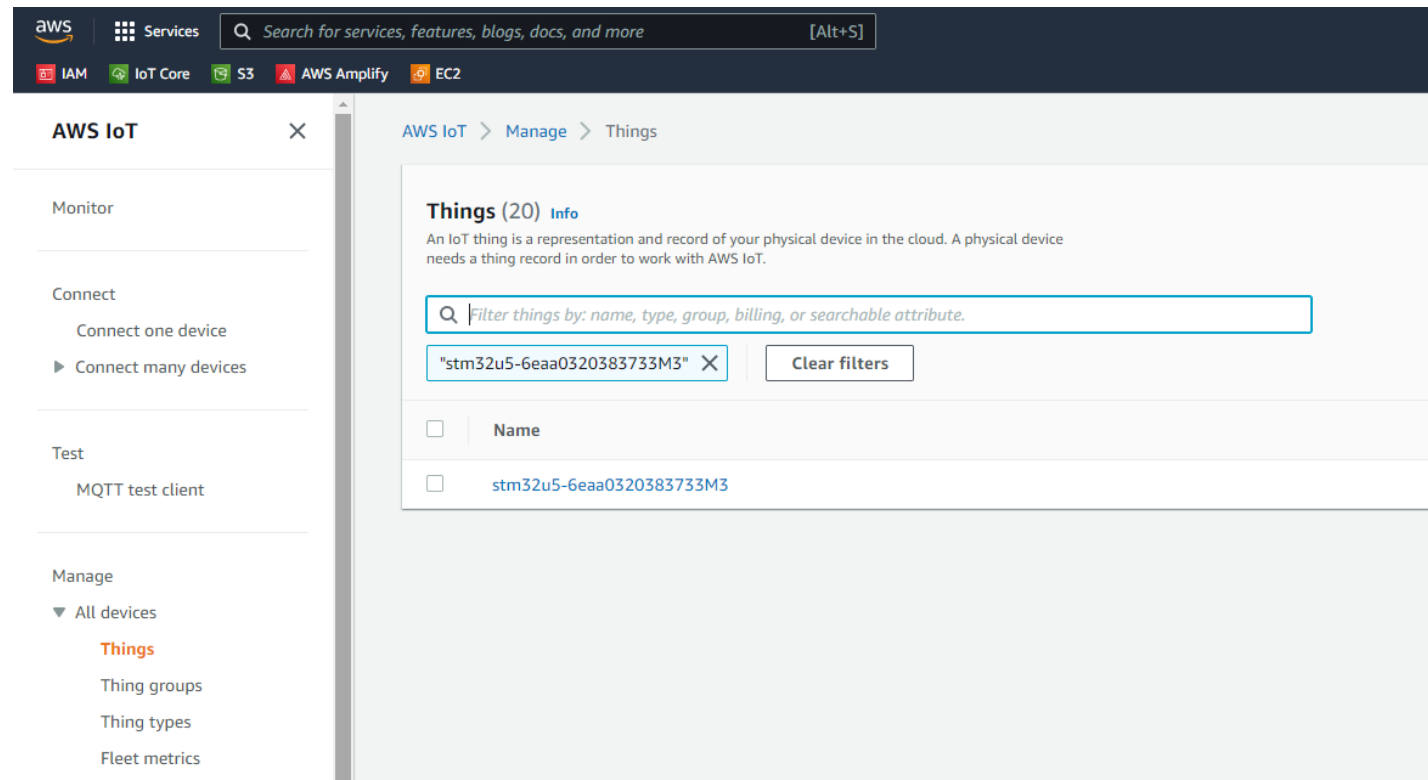
Copy your device name

- Copy your device name from the dashboard



Find your device in AWS IoT Core

- Open AWS IoT Core, select Manage → All devices → Things then paste your device name in the search bar
- Click on your device name



Explore your device properties

aws Services Search for services, features, blogs, docs, and more [Alt+S]

IAM IoT Core S3 AWS Amplify EC2

AWS IoT X

Monitor

Connect

- Connect one device
- ▶ Connect many devices

Test

- MQTT test client

Manage

- ▼ All devices
 - Things**
 - Thing groups
 - Thing types
 - Fleet metrics
- ▶ Remote actions
- ▶ Message Routing
 - Retained messages
- ▶ Security

Automate your security audit by enabling daily checks on your fleet from AWS IoT Device Defender. The audit evaluates your IoT configurations against security best practices, checks for identities and access policies. [View pricing](#) [Learn more](#)

stm32u5-6eaa0320383733M3 [Info](#)

Thing details

Name	stm32u5-6eaa0320383733M3	Type	-
ARN	arn:aws:iot:us-west-1:006151905315:thing/stm32u5-6eaa0320383733M3	Billing group	-

Attributes **Certificates** Thing groups Device Shadows Interact Activity Jobs Alarms Defender metrics

Certificates (4) [Info](#)

The device certificates attached to this thing resource.

Find certificates

<input type="checkbox"/>	Certificate ID	Status
<input type="checkbox"/>	2f8263816759d87c329bc393795c82021937b3d4f362c04ae39b9c5f41162...	Active

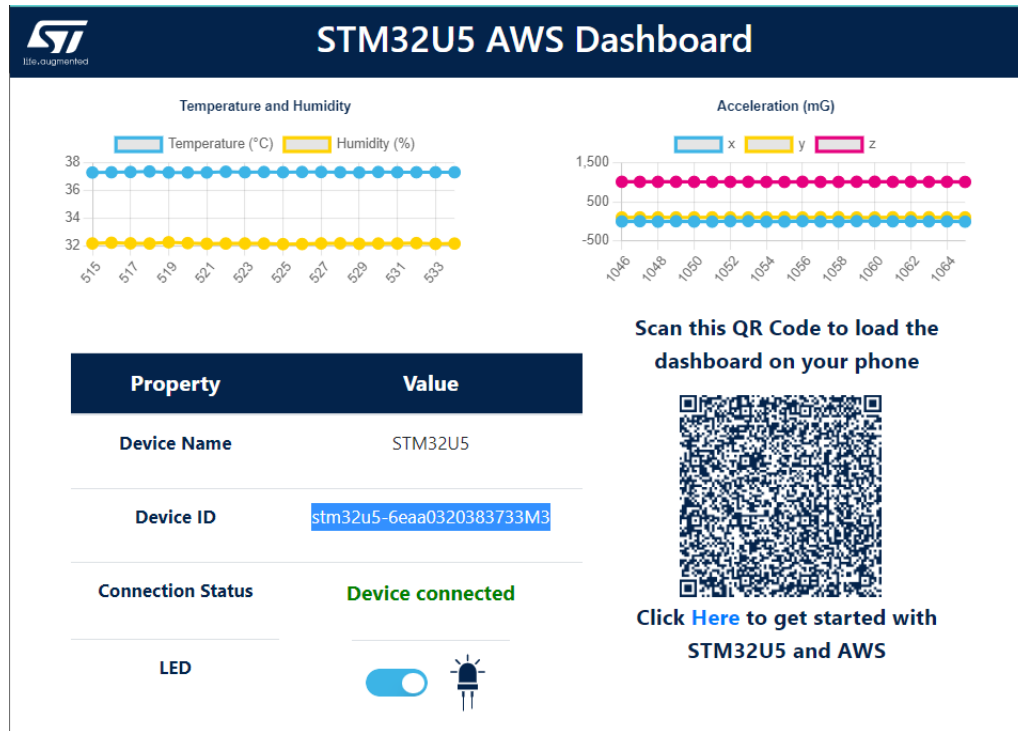
Select Activity

- Select the activity tab and then click on MQTT test client

The screenshot displays the AWS IoT console interface. On the left, a navigation sidebar lists various IoT-related actions: Monitor, Connect (with sub-options 'Connect one device' and 'Connect many devices'), Test (with 'MQTT test client'), and Manage (with a dropdown for 'All devices' and sub-items 'Things', 'Thing groups', 'Thing types', 'Fleet metrics', 'Remote actions', 'Message Routing', 'Retained messages', and 'Security'). The 'Things' item is highlighted. The main content area shows the details for a specific IoT device, 'stm32u5-6eaa0320383733M3'. At the top of this section is a blue banner indicating 'IoT security audit is off' with an 'Automate IoT security audit' button. Below this, the device name is displayed with 'Edit' and 'Delete' buttons. A 'Thing details' section provides information such as Name, ARN, Type, and Billing group. A horizontal tab bar at the bottom of the device details section includes 'Attributes', 'Certificates', 'Thing groups', 'Device Shadows', 'Interact', 'Activity' (which is selected and highlighted in orange), 'Jobs', 'Alarms', and 'Defender metrics'. The 'Activity' tab shows 'Activity (0)' with an 'Info' link and a description: 'Lists the most recent MQTT messages related to Device Shadow activity since you opened the thing details page. To see more messages related to this activity, choose the MQTT test client button.' A 'Clear' button and an 'MQTT test client' button with an external link icon are also present.

Check the shadow messages

- On the dashboard, toggle the LED and observe the messages exchanged on the MQTT Test client



Subscribe

Subscriptions

- \$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /update/+
- \$aws/things/stm32u5-6eaa0320383733M3/shadow/+ /rejected
- \$aws/events/presence/+ /stm32u5-6eaa0320383733M3
- \$aws/events/subscriptions/+ /stm32u5-6eaa0320383733M3
- \$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /documents
- \$aws/things/stm32u5-6eaa0320383733M3/shadow/+ /accepted
- \$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /delete/+
- \$aws/things/stm32u5-6eaa0320383733M3/shadow/update/documents

\$aws/things/stm32u5-6eaa0320383733M3/shadow/update/documents

Pause Clear Export Edit

► \$aws/things/stm32u5-6eaa0320383733M3/shadow/update/documents September 27, 2022, 12:14:09 (UTC-0700)

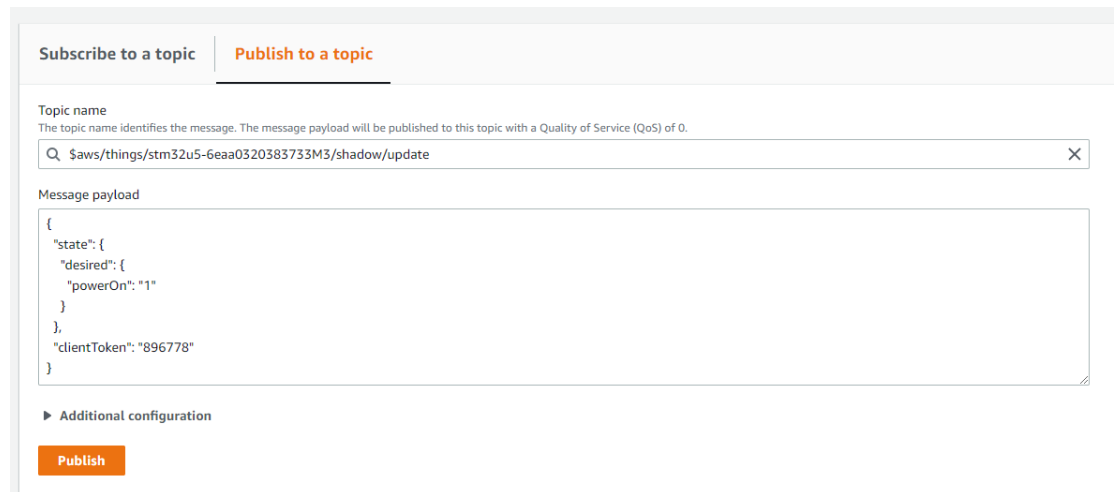
▼ \$aws/things/stm32u5-6eaa0320383733M3/shadow/update/documents September 27, 2022, 12:14:09 (UTC-0700)

```
{
  "previous": {
    "state": {
      "desired": {
        "powerOn": "0"
      },
      "reported": {
        "powerOn": 0
      }
    },
    "metadata": {
      "desired": {
        "powerOn": {
          "timestamp": 1664305518
        }
      },
      "reported": {
        "powerOn": 0
      }
    }
  }
}
```

Publish to your board shadow

- Select **Publish to topic**
- On the topic name put `$aws/things/<your board name>/shadow/update`
- Use the following json message as message payload

```
{
  "state": {
    "desired": {
      "powerOn": "1"
    }
  },
  "clientToken": "896778"
}
```



The screenshot shows the 'Publish to a topic' interface in the AWS IoT console. It has two tabs: 'Subscribe to a topic' and 'Publish to a topic', with the latter being selected. Below the tabs, there is a 'Topic name' field with a placeholder text 'The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.' The field contains the text '\$aws/things/stm32u5-6eaa0320383733M3/shadow/update'. Below this is a 'Message payload' text area containing a JSON object: { "state": { "desired": { "powerOn": "1" } }, "clientToken": "896778" }. At the bottom, there is an 'Additional configuration' section and a 'Publish' button.

- Change the `powerOn` desired state from 0 to 1 and from 1 to 0
- Observe the LED on your board and on the dashboard

Lab 5: Telemetry messages

Subscribe to telemetry topics

- Select **Subscribe to topic**
- On the topic name put `<your board name>/#`
- Observe the telemetry data

You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topics to communicate their state to AWS IoT. AWS IoT also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages to topics by using the MQTT test client.

Subscribe to a topic | **Publish to a topic**

Topic filter [Info](#)
The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.

stm32u5-6eaa0320383733M3/#

► Additional configuration

Subscribe

Subscriptions

\$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /update/+	♡ ✕
\$aws/things/stm32u5-6eaa0320383733M3/shadow/+ /rejected	♡ ✕
\$aws/events/presence/+ /stm32u5-6eaa0320383733M3	♡ ✕
\$aws/events/subscriptions/+ /stm32u5-6eaa0320383733M3	♡ ✕
\$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /documents	♡ ✕
☑ \$aws/things/stm32u5-6eaa0320383733M3/shadow/+ /accepted	♡ ✕
\$aws/things/stm32u5-6eaa0320383733M3/shadow/name/+ /delete/+	♡ ✕

stm32u5-6eaa0320383733M3/# Pause Clear Export Edit

▼ stm32u5-6eaa0320383733M3/env_sensor_data September 27, 2022, 12:26:56 (UTC-0700)

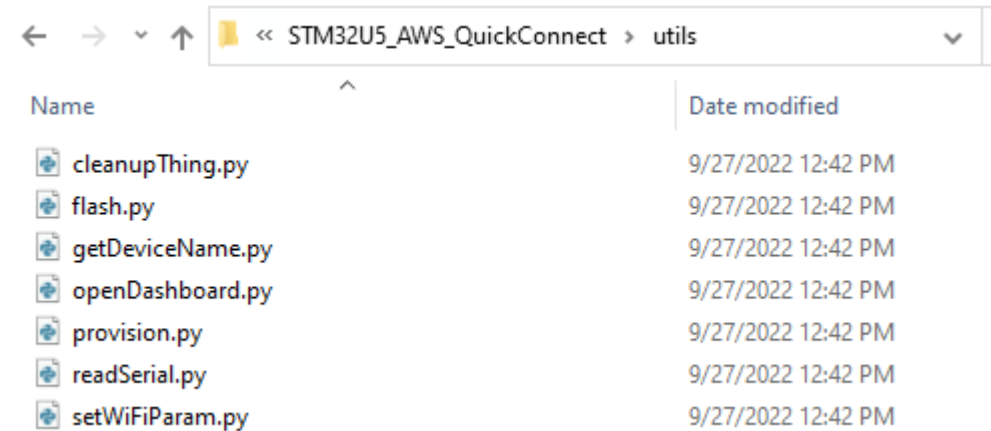
```
{
  "temp_0_c": 37.654301,
  "rh_pct": 31.729755,
  "temp_1_c": 37.290001,
  "baro_mbar": 1005.970703
}
```

September 27, 2022, 12:26:56 (UTC-0700)

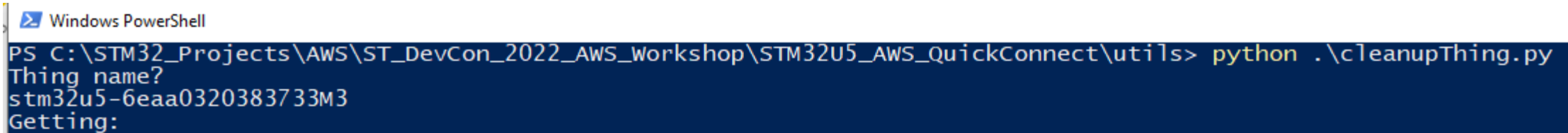
Lab 6: Delete your device

Delete your device

- Navigate to `STM32U5_AWS_QuickConnect\utils`
- Type `python .\cleanupThing.py`
- Then type your thing name



Name	Date modified
cleanupThing.py	9/27/2022 12:42 PM
flash.py	9/27/2022 12:42 PM
getDeviceName.py	9/27/2022 12:42 PM
openDashboard.py	9/27/2022 12:42 PM
provision.py	9/27/2022 12:42 PM
readSerial.py	9/27/2022 12:42 PM
setWiFiParam.py	9/27/2022 12:42 PM



```
PS C:\STM32_Projects\AWS\ST_DevCon_2022_AWS_Workshop\STM32U5_AWS_QuickConnect\utils> python .\cleanupThing.py
Thing name?
stm32u5-6eaa0320383733M3
Getting:
```

Our technology starts with You



Find out more at www.st.com

© STMicroelectronics - All rights reserved.

ST logo is a trademark or a registered trademark of STMicroelectronics International NV or its affiliates in the EU and/or other countries.

For additional information about ST trademarks, please refer to www.st.com/trademarks.

All other product or service names are the property of their respective owners.



life.augmented