**Air Quality Project**

**Prerequisites**



March 24th, 2022

# **Introduction**

In the scope of the Air Quality project, you will work on several different security parts of the system, both Backend and Frontend. The goal of this document is to provide a list – as exhaustive as possible – of software and drivers you will need to previously install on your computer before the beginning of the session.

# **1. Repository**

First, you need to access to the repository to retrieve all the Backend and Frontend code

<https://github.com/ChristopheLaurent/AirQuality>

If you have any issue to connect to this repository, please contact Christophe Laurent, the owner of this repository to grant access.

* Create a new development folder on your computer (as your convenience) as *C:\<Project>\AirQuality*
* Download the **AirQualityBackend, AirQualitySensor** and **Model3D** from the repository and copy it in the *C:\<Project>\AirQuality* folder.

# **2. Backend**

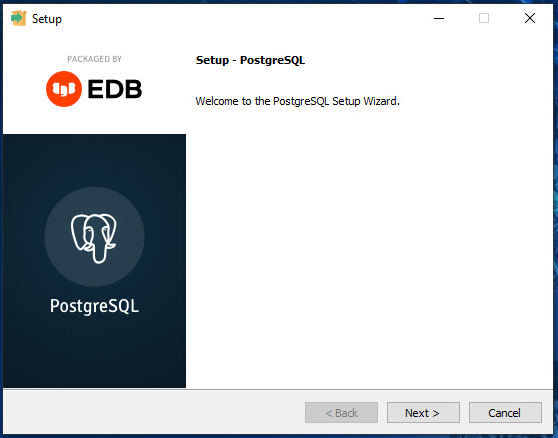
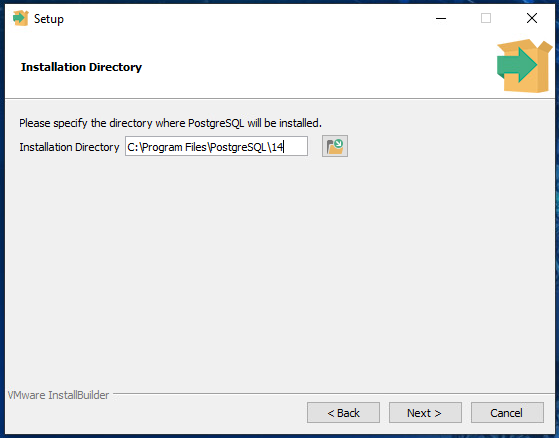
The backend is based on Python code, associated to MQTT protocol and the data will be stored in a PostGreSQL database. You will need to install them on your computer.

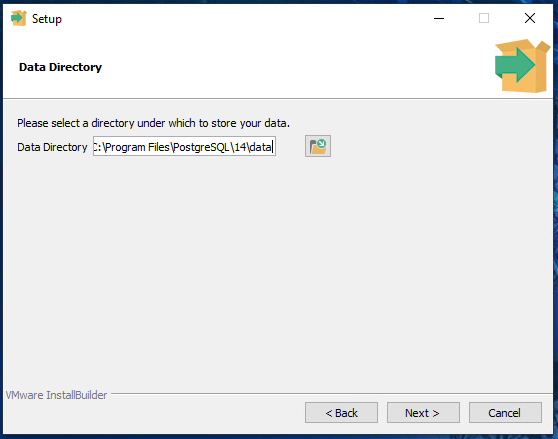
## PostGreSQL Database

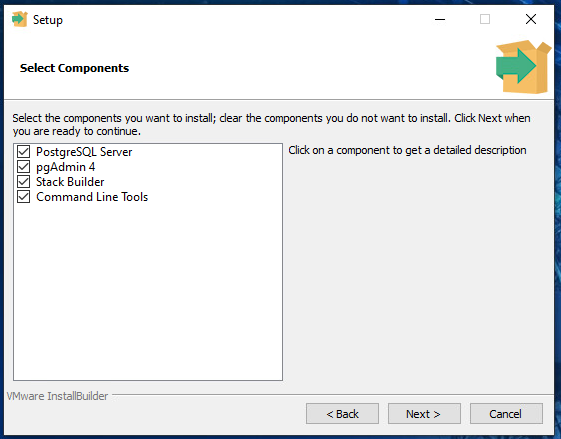
Download the last release of PostGreSQL database (14.2) on

<https://www.postgresql.org/download/windows/>

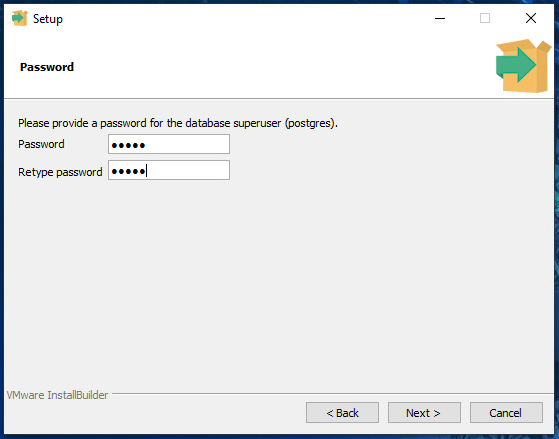
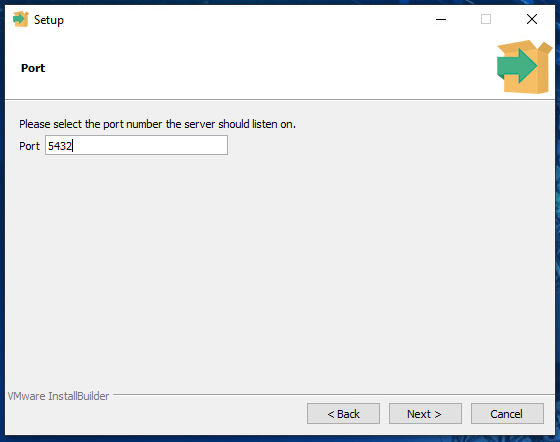
and follow the wizard (run as admin on your computer if needed).

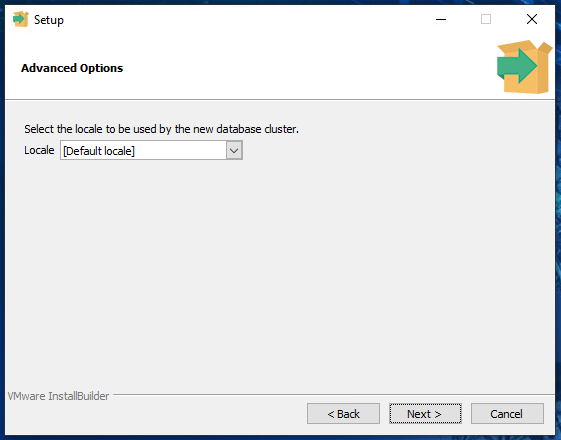
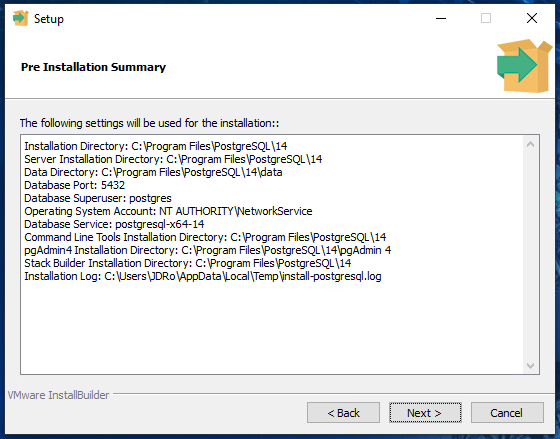


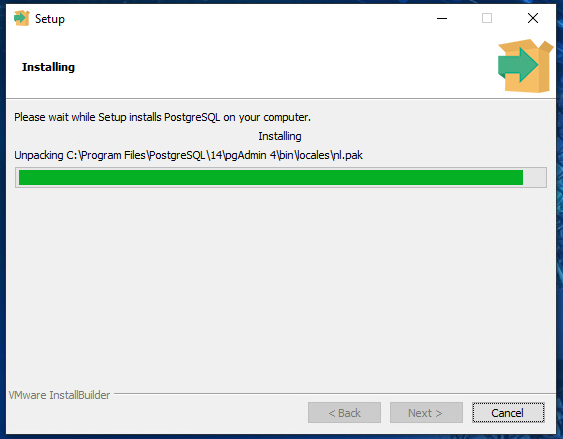


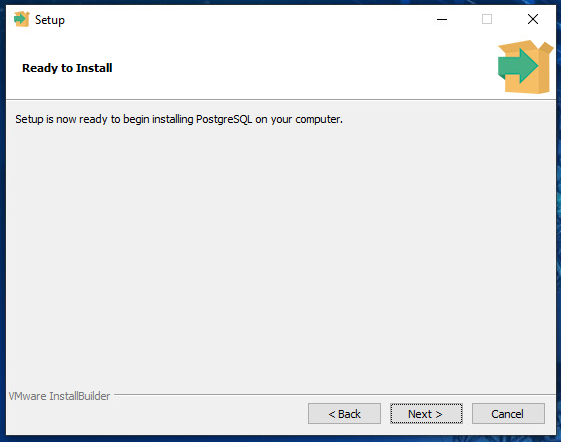


For the superuser (postgres) password, enter **admin**

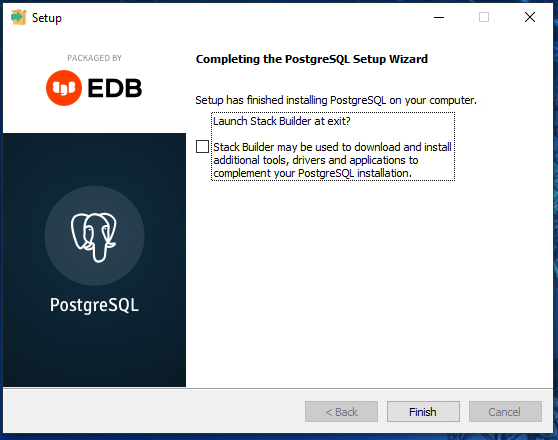








Untick **Launch StackBuilder at exit** and click on **Finish**

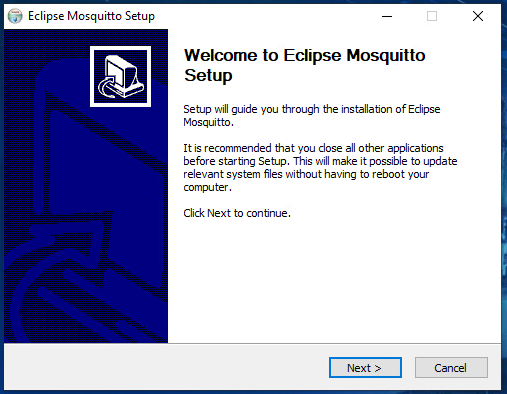
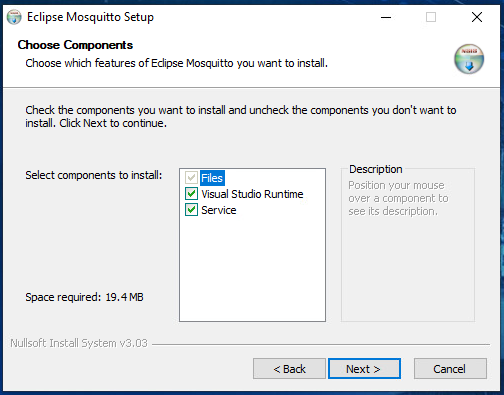


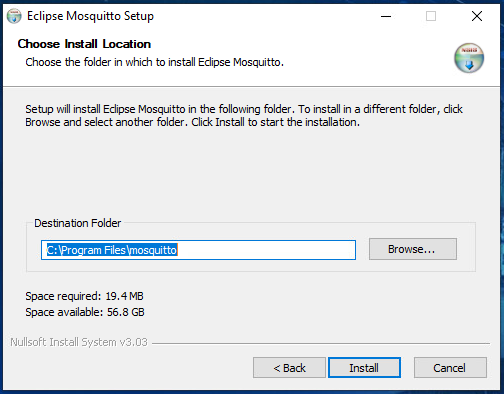
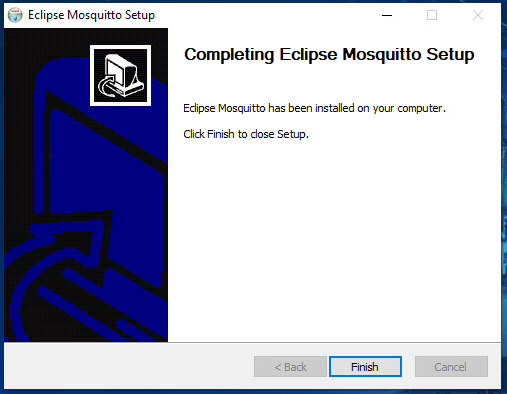
Using **pgadmin**, create a new database named “**AirQuality**”.

## 2.2 Mosquitto Broker

Mosquitto broker is a message broker that implements the MQTT protocols

Download it from [https://mosquitto.org/download/](https://mosquitto.org/download/a) and install it (run as admin on your computer if needed).





Reboot your computer.

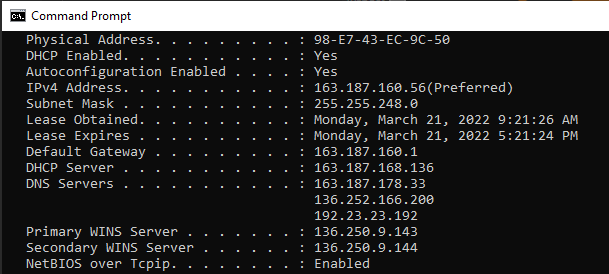
After the reboot start the **Task Manager** and confirm that the *mosquitto.exe* process is running.



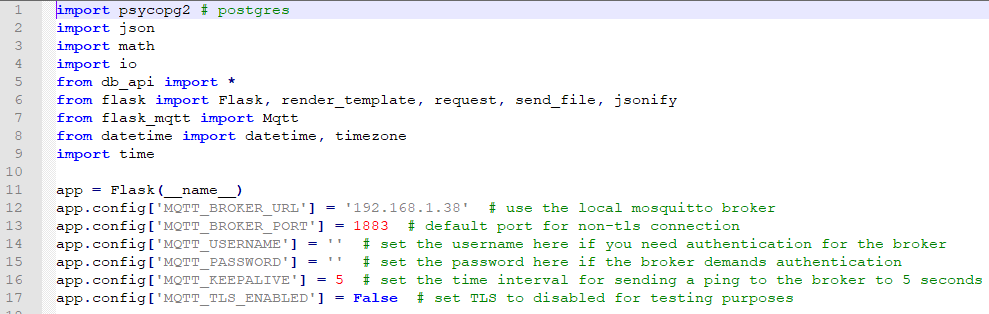
## 2.3 Setting of the IP address in several files

At his step you must know the IP address of the computer where Mosquitto broker is running.

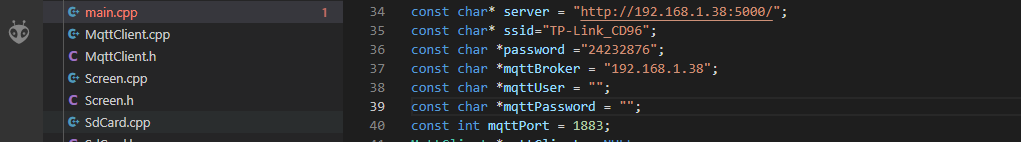
* Open a Dos window
* Cmd
* Ipconfig /all
* Retrieve the IP address of the computer (as in the following example)



* Edit *AirQualityBackend\app.py* file and set the correct IP address



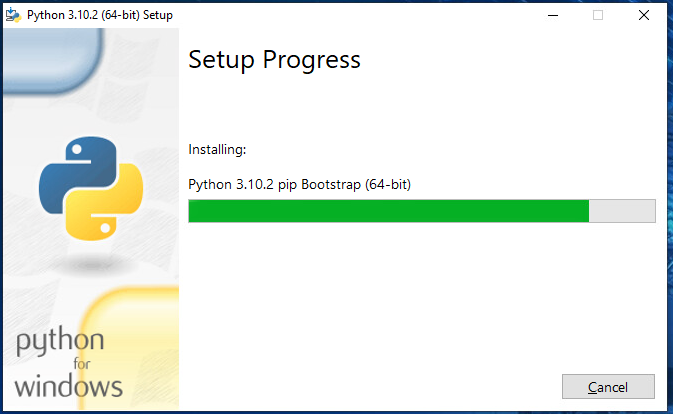
* Edit *AirQualitySensor\src\Main.cpp* file and set the settings: **server**, **ssid**, **password**, **mqttBroker**



## 2.4 Python

The last release of python (3.10.2) is available on <https://www.python.org/downloads/>

During the installation, tick the **Add Python 3.10 to PATH**



Once you have installed python, you will install some other packages.

For your information these are the needed package

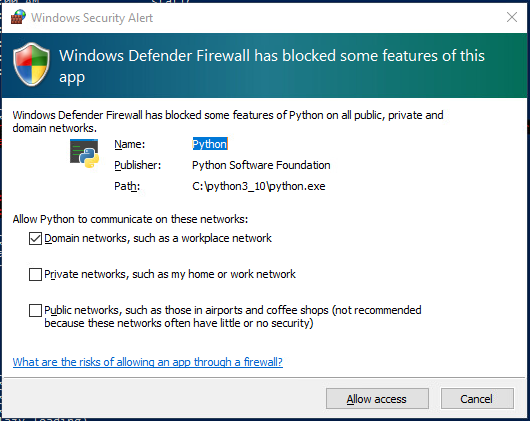
<https://flask.palletsprojects.com/en/2.0.x/installation/#install-flask>

<https://pypi.org/project/psycopg2/>

<https://flask-mqtt.readthedocs.io/en/latest/>

* Run (as admin) Windows Powershell
* cd *C:\<Project>\AirQuality/AirQualityBackend*
* Set-ExecutionPolicy Unrestricted -Force
* pip install virtualenv
* virtualenv venv
* venv\Scripts\activate
* pip install Flask
* pip install psycopg2
* pip install flask-mqtt
* python init\_db.py
* $env:FLASK\_APP = "app"
* flask run --host=0.0.0.0

if the following window appears, click Allow access

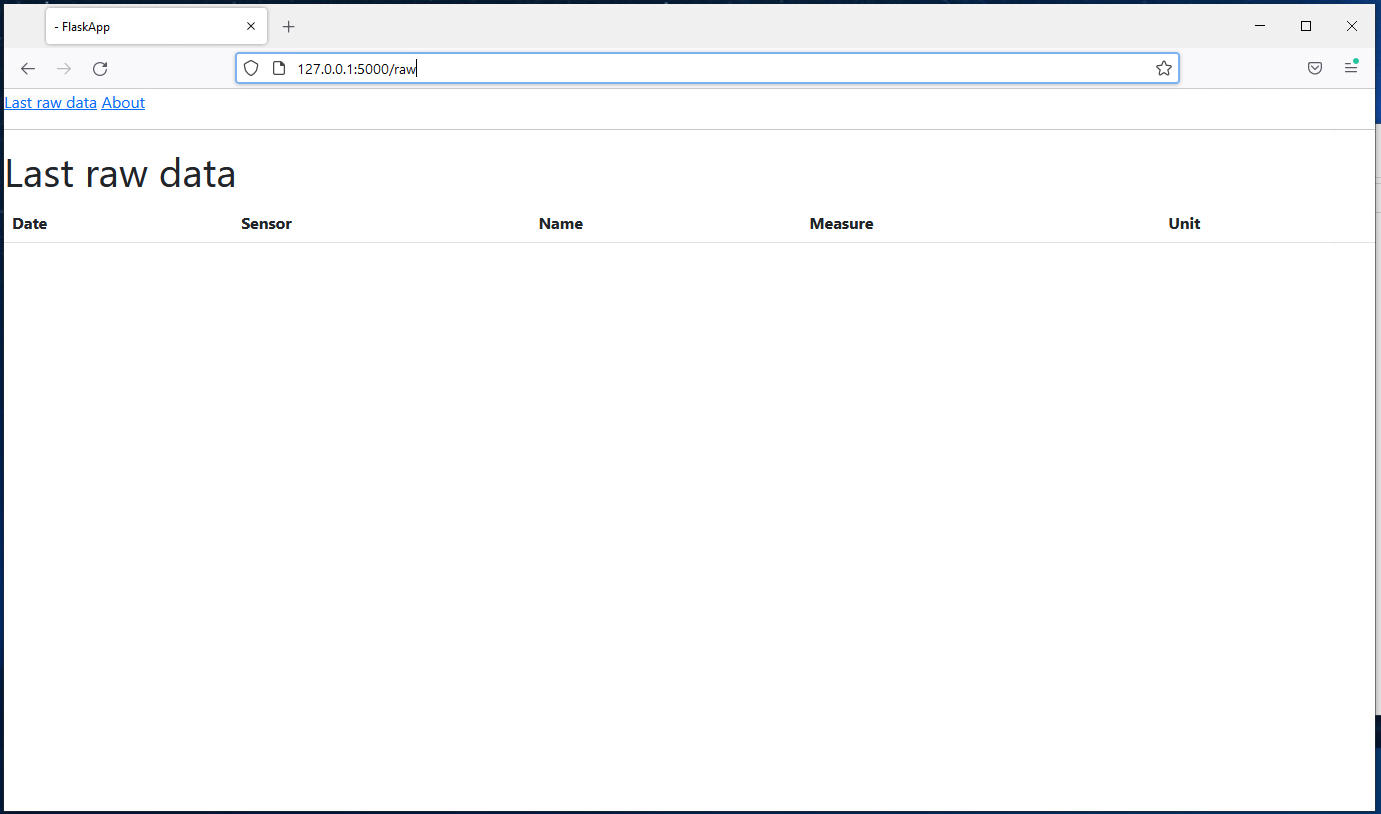


## 2.5 Check the development environment

Once you have installed all the previous software you can easily check if all your environment is properly installed.

* Open your favorite browser and go to <http://127.0.0.1:5000/raw>

If all is running well, you must have the following page displayed.



The Backend is now operational. You will install and configure the Frontend part.

# **3. Frontend**

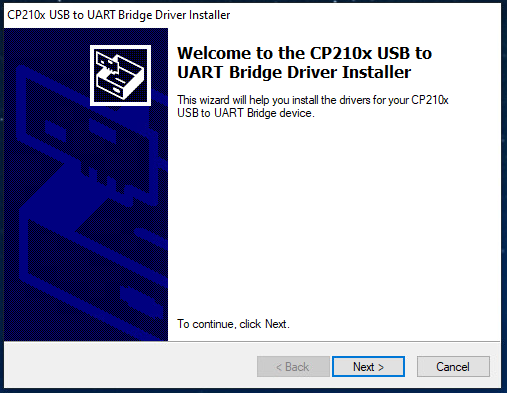
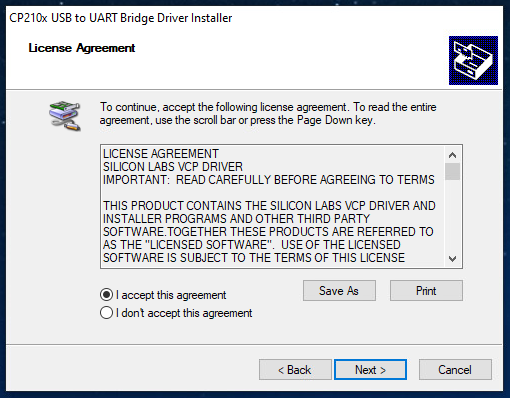
On the Frontend side, you need to install PlatformIO IDE based on Visual Studio Code and a Windows driver to provide the communication between the software and the device.

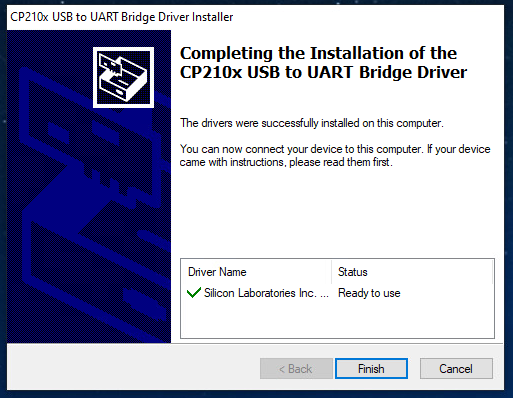
## 3.1 Driver

You need first manually install the **CP210x Driver** on your computer to access to the EPaper device from PlatformIO.

Go to <https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers>

In Downloads -> CP210x Windows Drivers





## 3.2 Visual Studio Code

Download the Visual Studio Code from <https://code.visualstudio.com/download> and install it.

## 3.3 Platform IO

This IDE is an extension of Visual Studio Code previously installed.

Go to <https://platformio.org/>   
Click on **Get it Now!** and follow the instructions to install this extension under Visual Studio Code.

In Visual Studio Code 🡪 View 🡪 Extensions 🡪PlatformIO IDE

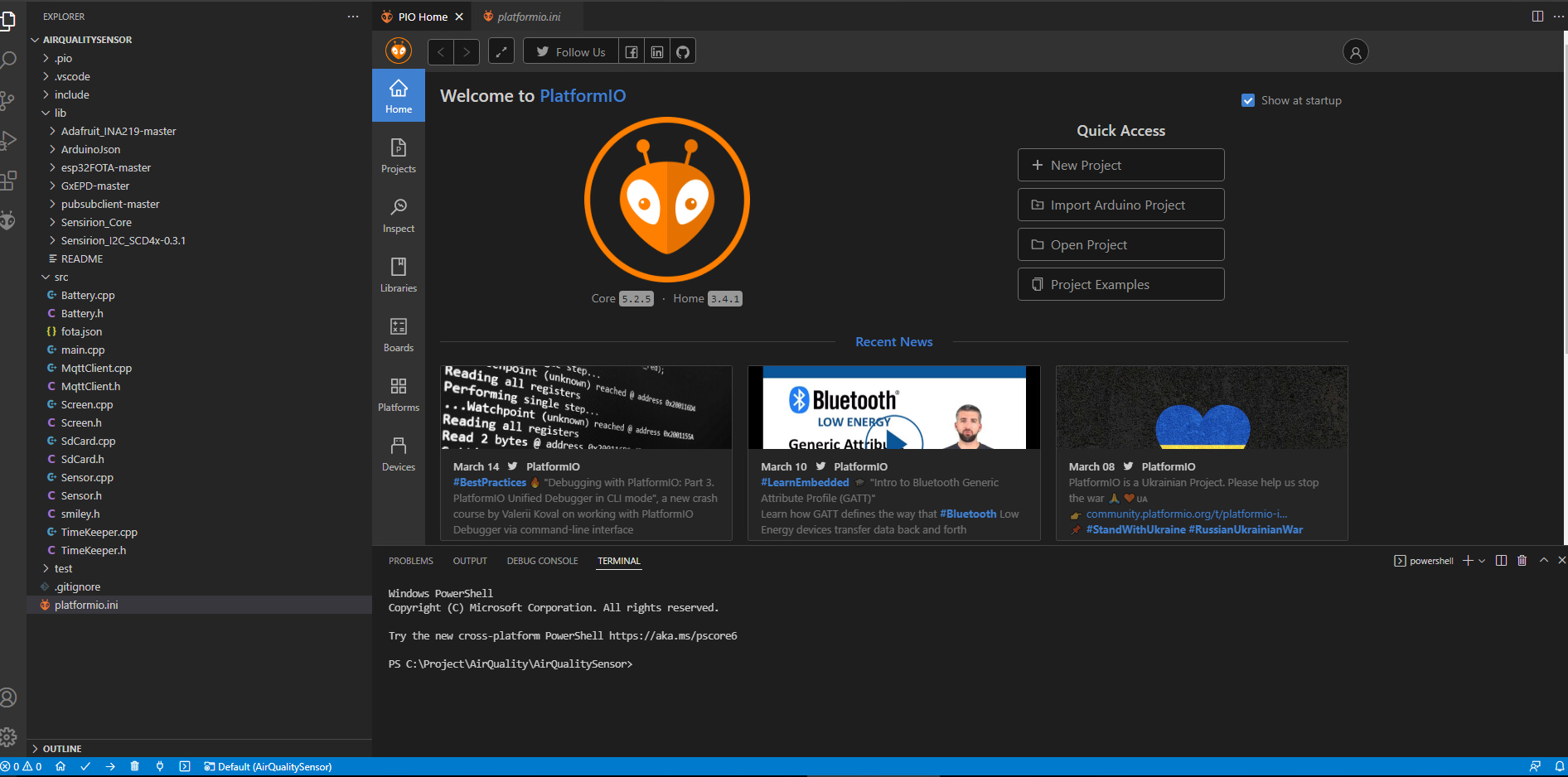
Restart Visual Studio Code

File 🡪 Open Folder 🡪 *C:\<Project>\AirQuality/AirQualitySensor* 🡪 Select Folder

At the bottom, click on **PlatformIO: Home** icon

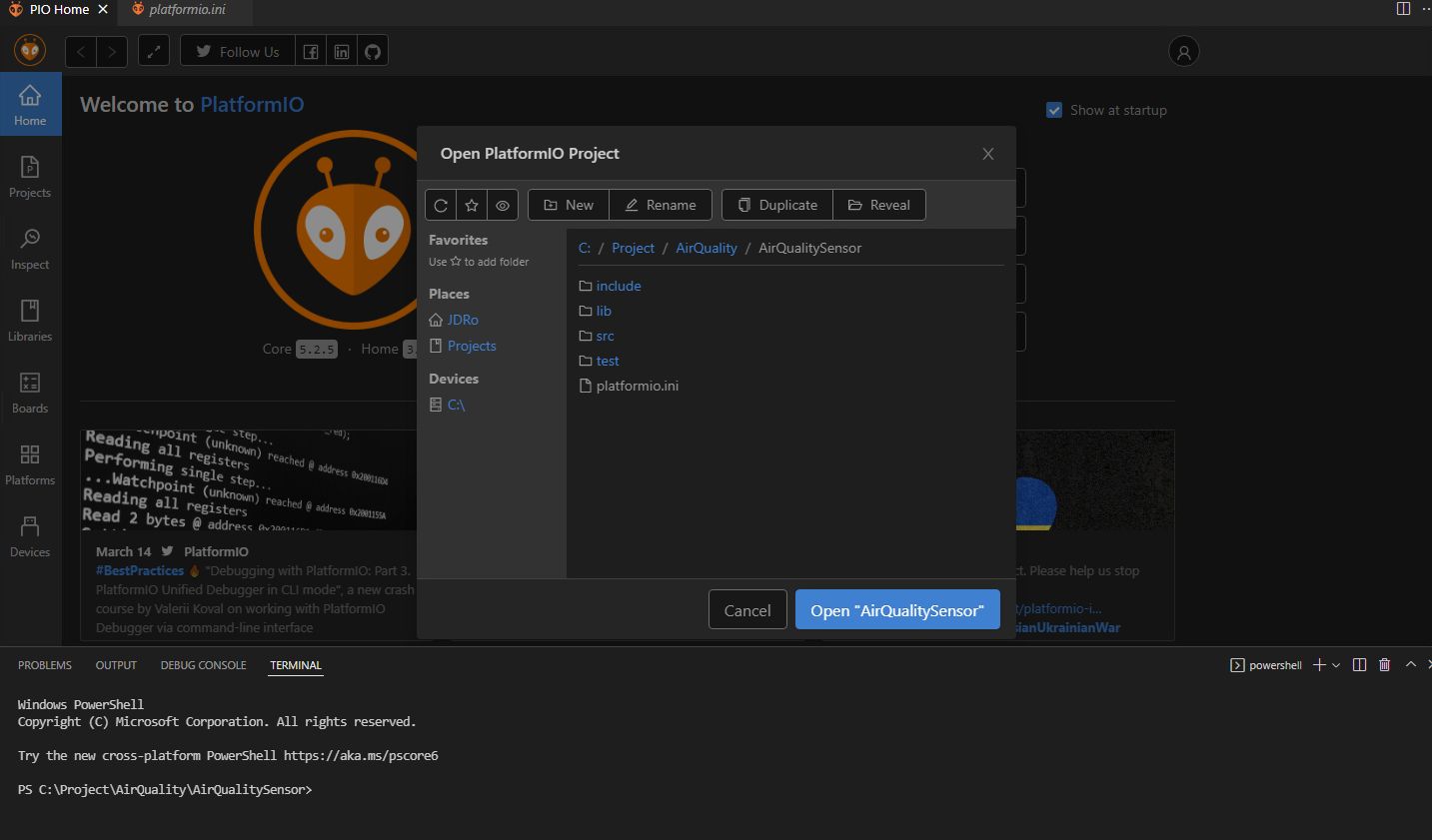


You have this window



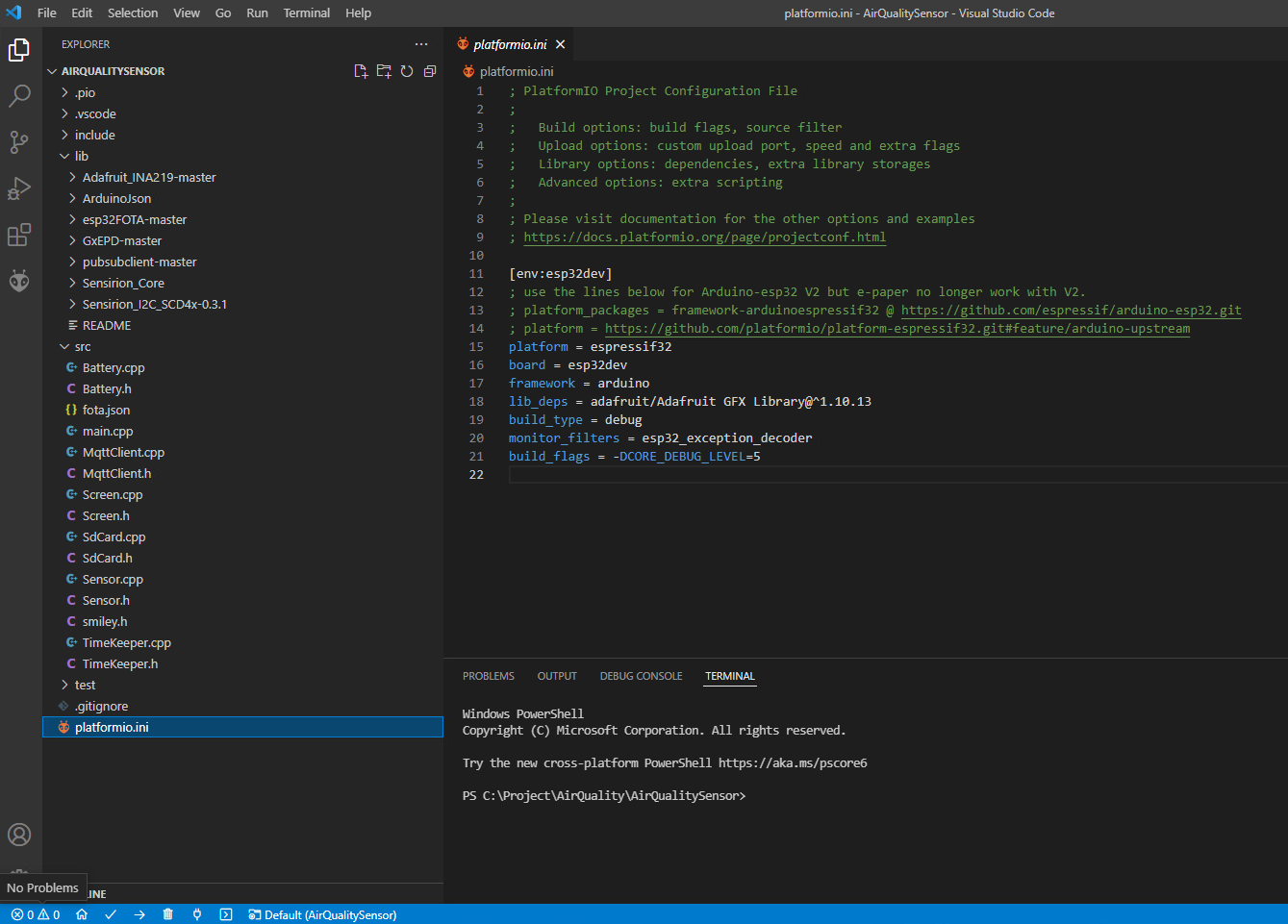
Click on **Open Project**

Select *C:\<Project>\AirQuality/AirQualitySensor*



Click on **Open** **“AirQualitySensor”**

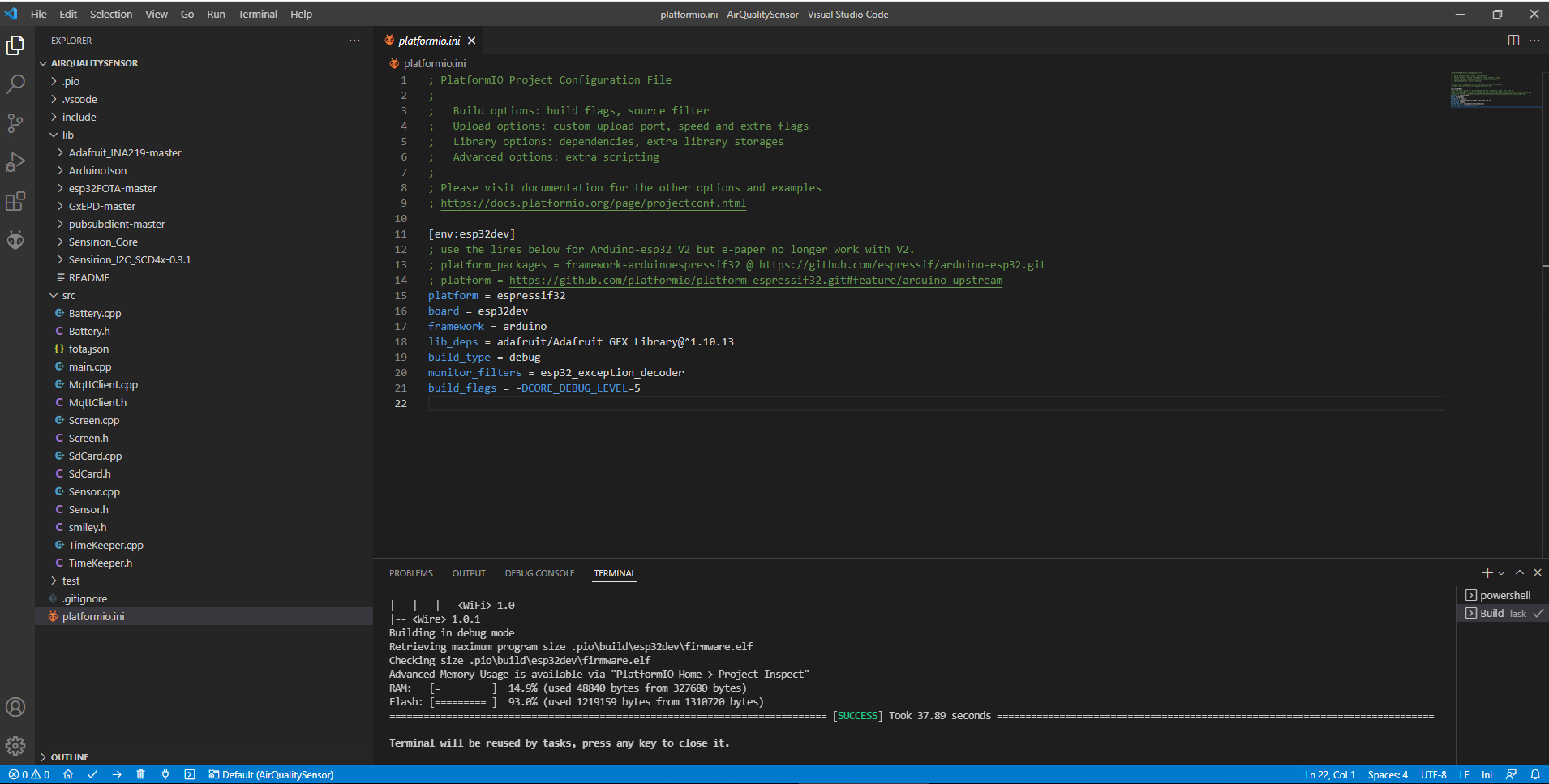
The platform.ini file is displayed



At the bottom, click on **PlatformIO: Build** icon



You have successfully built the project and you are ready to inject it in the ESP32



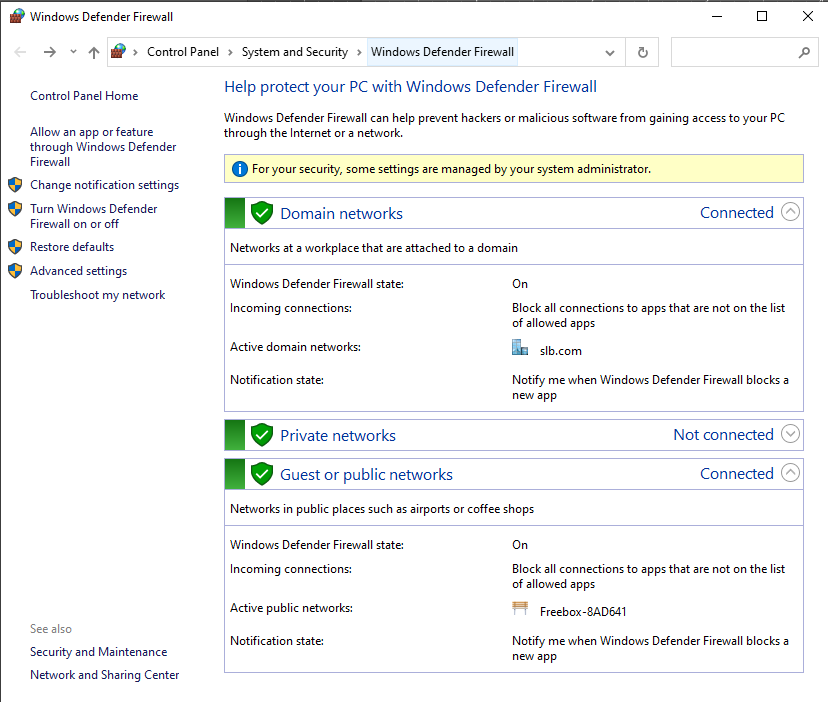
At the bottom, click on **PlatformIO: Upload** icon to upload the firmware in the device



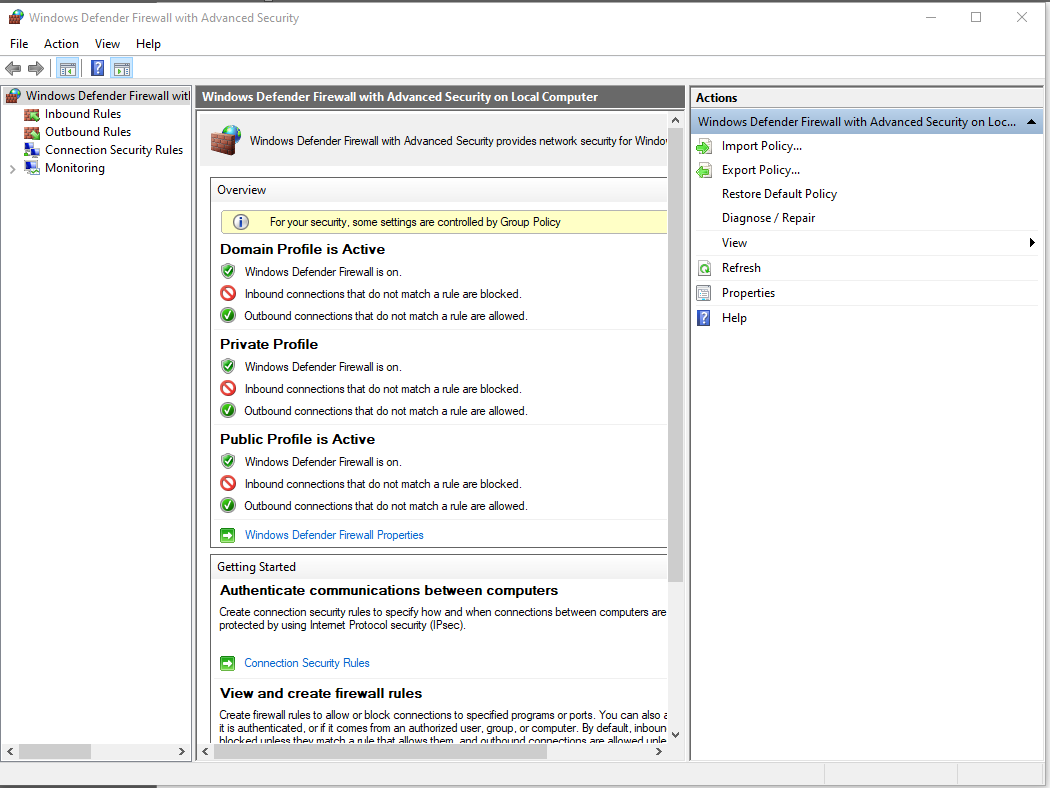
# **4. Firewall Configuration**

You must add some rules in your *Windows Firewall* to allow the communication from the Device that sending data by MQTT protocol to the MQTT broker located on your computer.   
Without these inbound and outbound rules, no data are received by your MQTT broker and no data is recorded in the database.

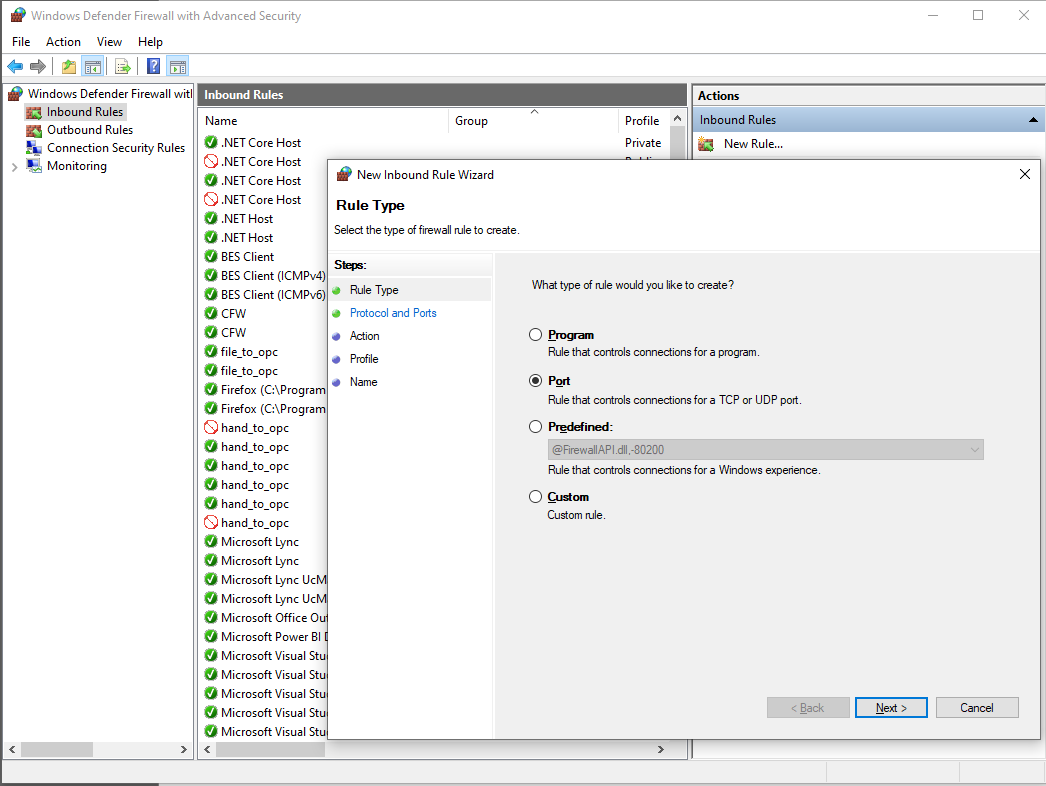
Go to **Windows Defender Firewall**



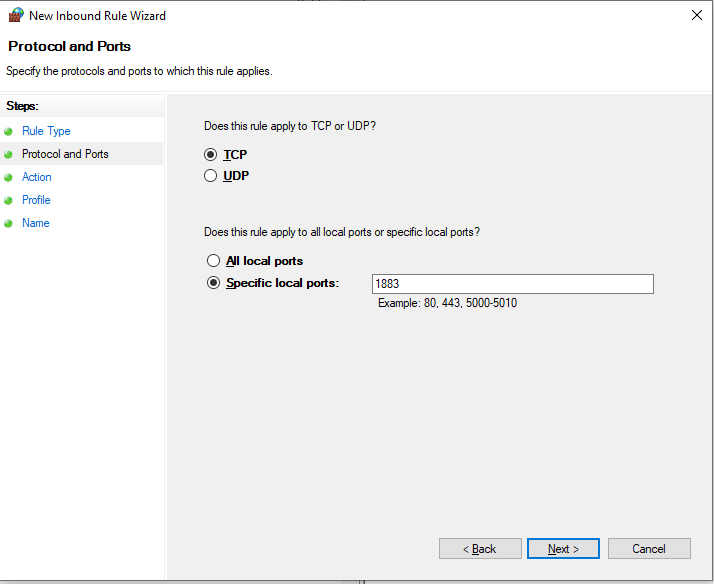
Click on **Advanced settings**



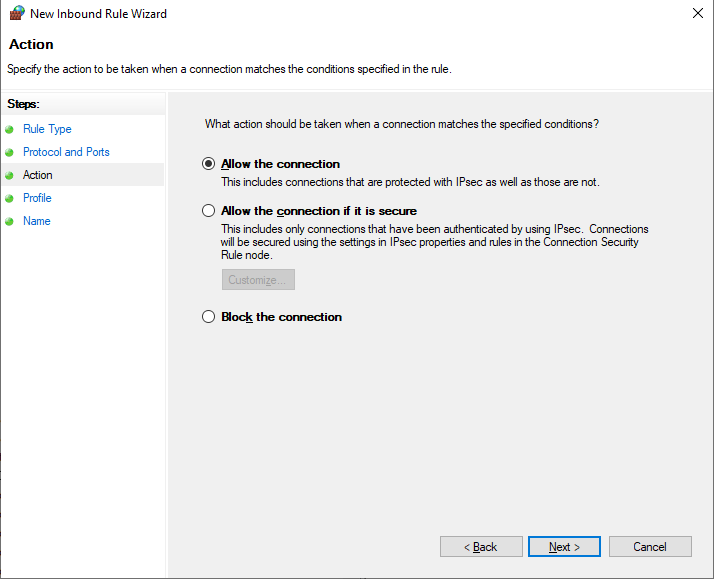
Click on **Inbound Rules** 🡪 **New Rule…** 🡪 Select **Port** 🡪 **Next**



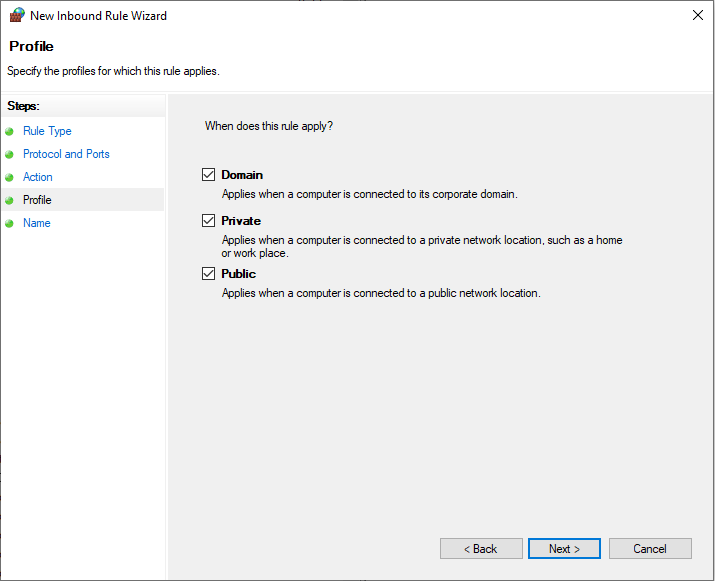
**TCP** and **Specific local ports**: and enter **1883** (this is the port used by MQTT) 🡪 **Next**



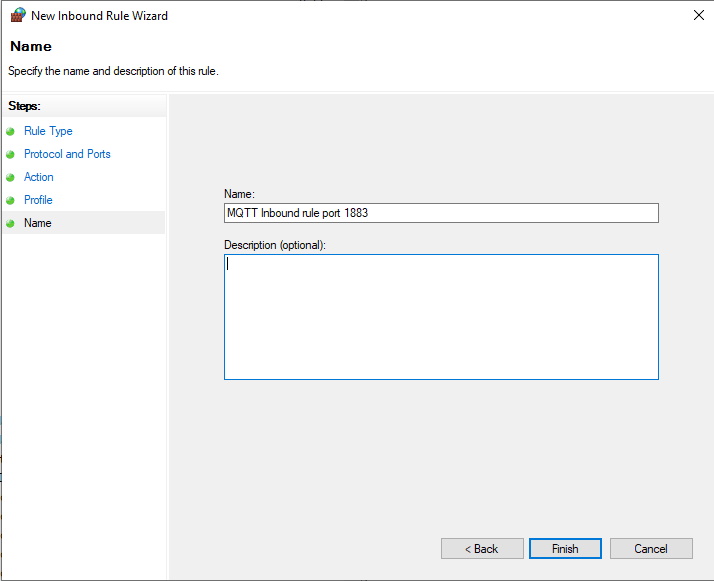
**Allow the connection** 🡪 **Next**



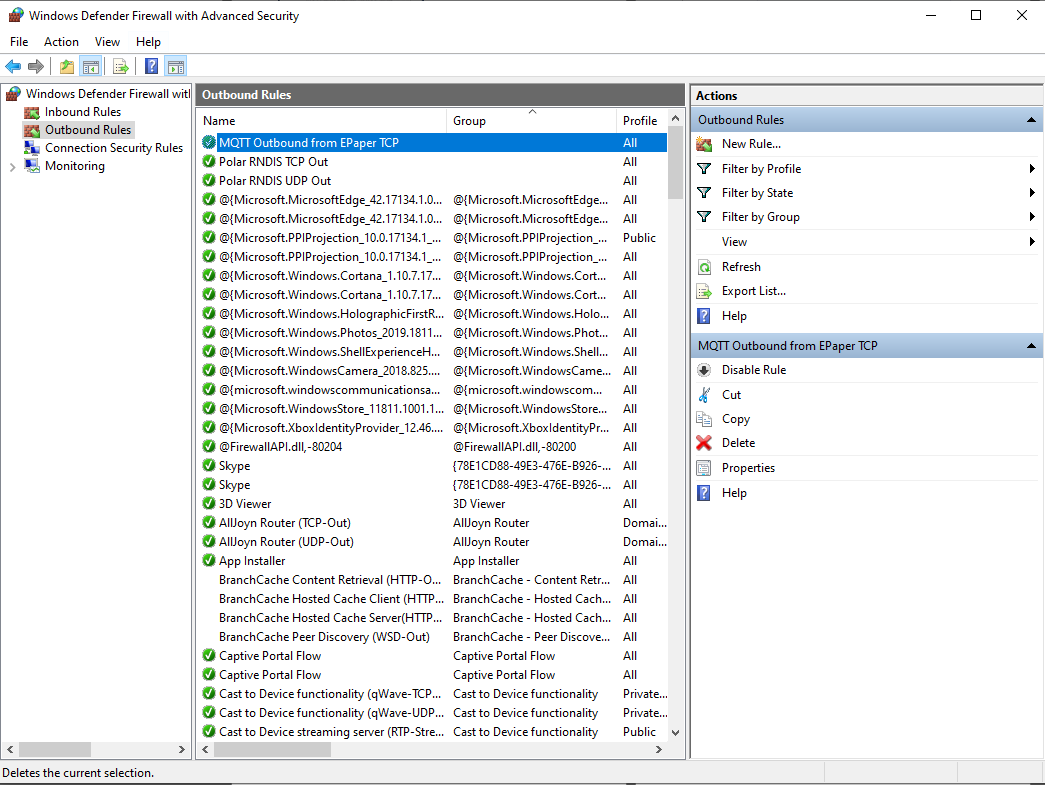
Select **Domain**, **Private** and **Public** 🡪 **Next**



Enter the name of the rule (as you want) 🡪 **Finish**

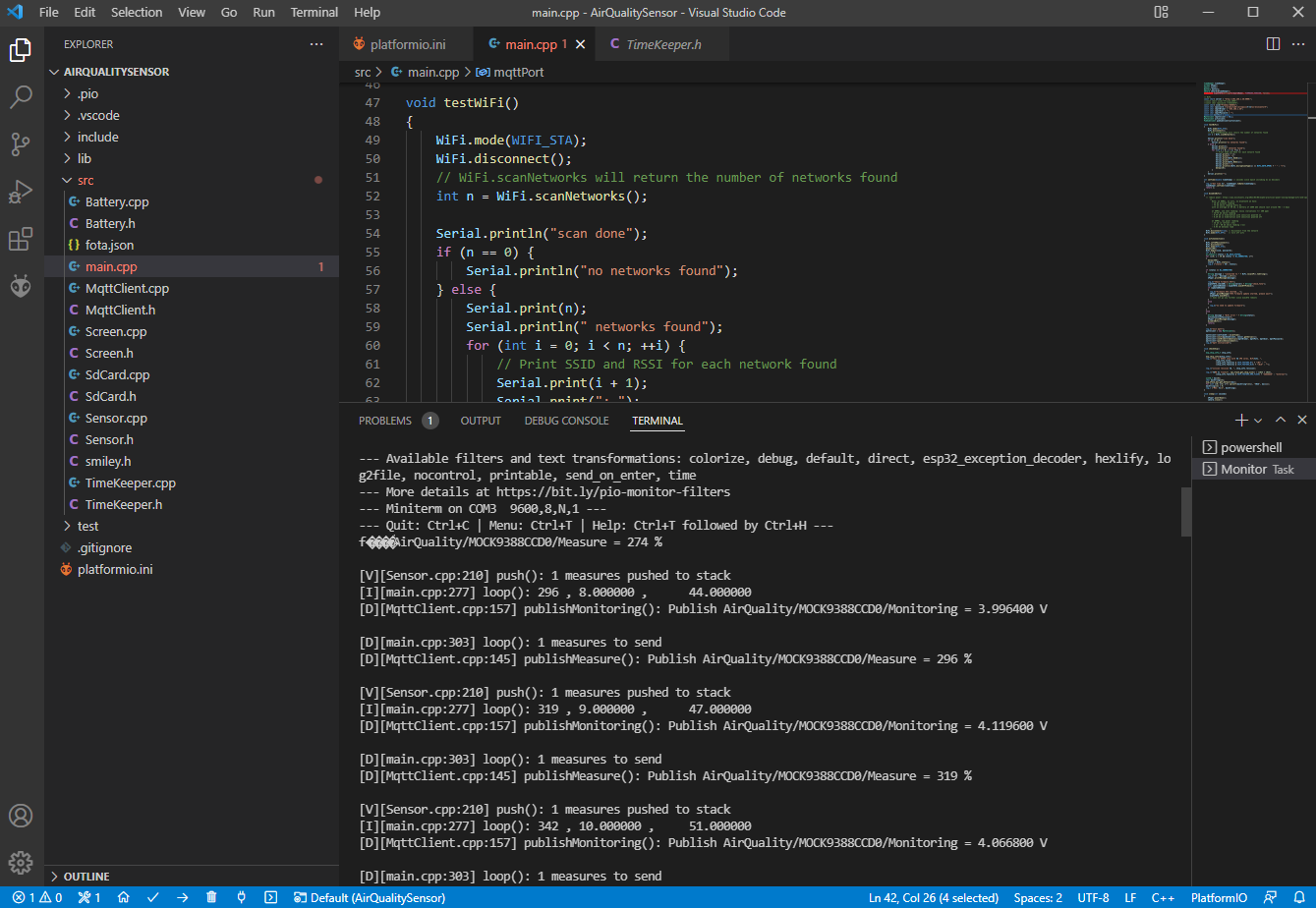


**Redo the same procedure to add an Outbound Rule on port 1883**



# **5. Tests phase**

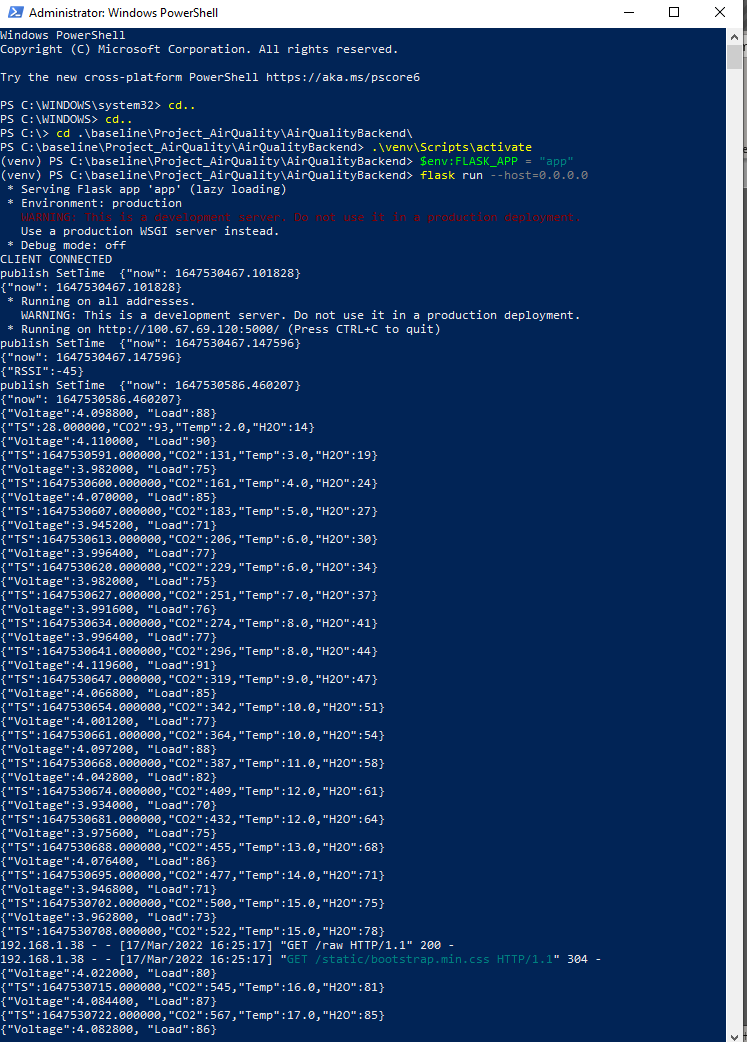
Once you are ready, click in Visual Studio Code the *power outlet* icon to activate the traces and see the communication messages



In the previous screenshot you can see the received data after the start of the device

Connect the device (by USB) to your computer and switch ON.

After few seconds, you can see in PowerShell window information received by Flask server and data received from the device (voltage , temperature, ….)



The highlighted block of data in the previous screenshot shows us the request from the browser to display the data in a webpage, result you can see just underneath

