Welcome!

In this page we will try to give you an example of a sensing service that publishes measurements to our platform and demonstrate how to deploy it, step by step!

For this particular demonstration we chose to deploy an **S-type service** (see *Services* section in *Home* Wiki) which utilises a **TEMPer** USB sensor connected to your machine, to get the *room temperature* as an *input* and display it in *Celsius* or *Fahrenheit* format as an *output*.

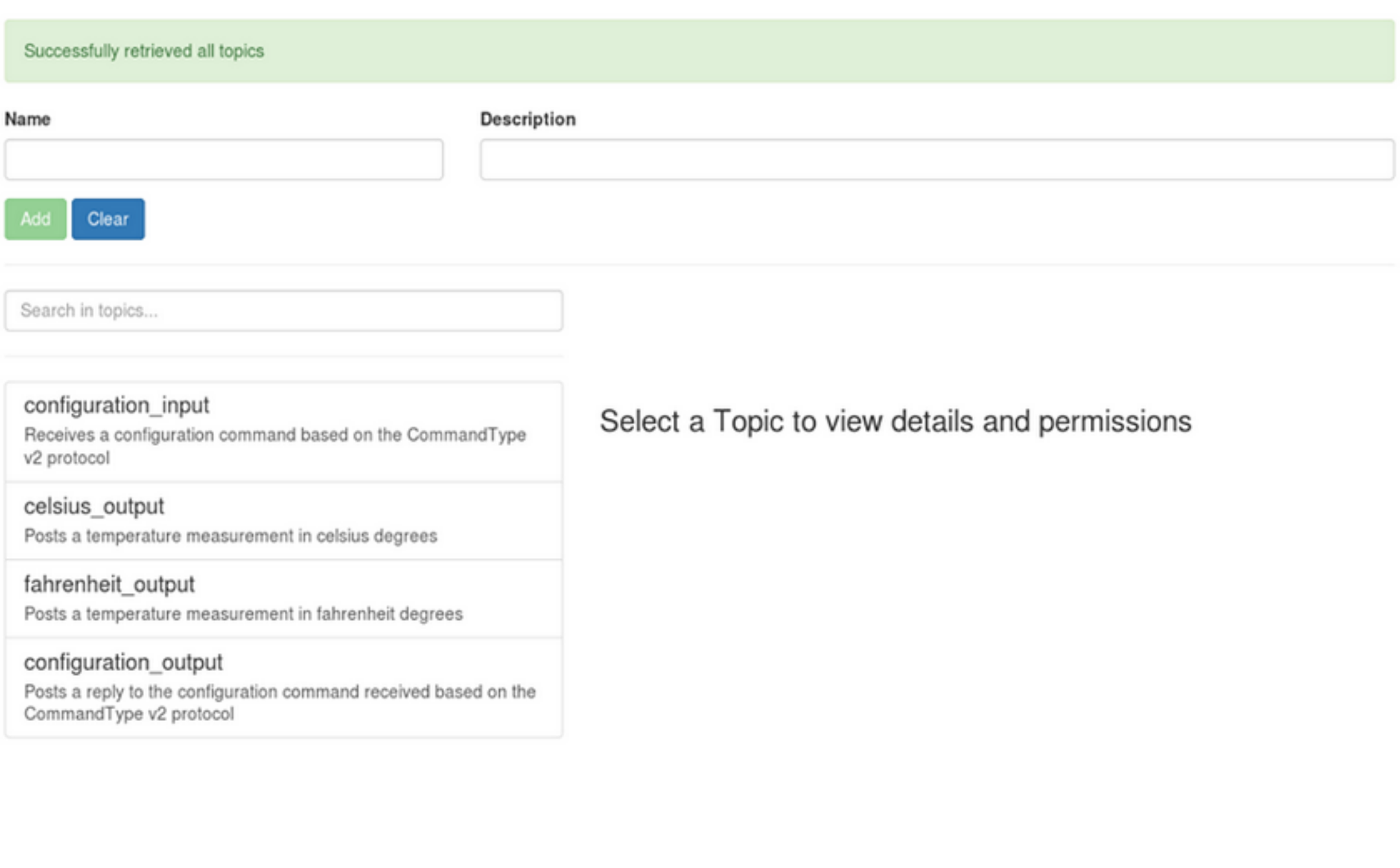
First things First!

Log in to **SYNAISTHISI** the way we described in the *Home* Wiki page and go to the **Topics** menu.

There, you should create the following 4 topics, exactly as described:

> **1st Topic:**
>
> - **Name:** configuration_input
> - **Description:** Receives a configuration command based on the CommandType v2 protocol
>
> **2nd Topic:**
>
> - **Name:** celsius_output
> - **Description:** Posts a temperature measurement in celsius degrees
>
> **3rd Topic:**
>
> - **Name:** fahrenheit_output
> - **Description:** Posts a temperature measurement in fahrenheit degrees
>
> **4th Topic:**
>
> - **Name:** configuration_output
> - **Description:** Posts a reply to the configuration command received based on the CommandType v2 protocol

*When you are finished, the **Topics** menu should look like:*



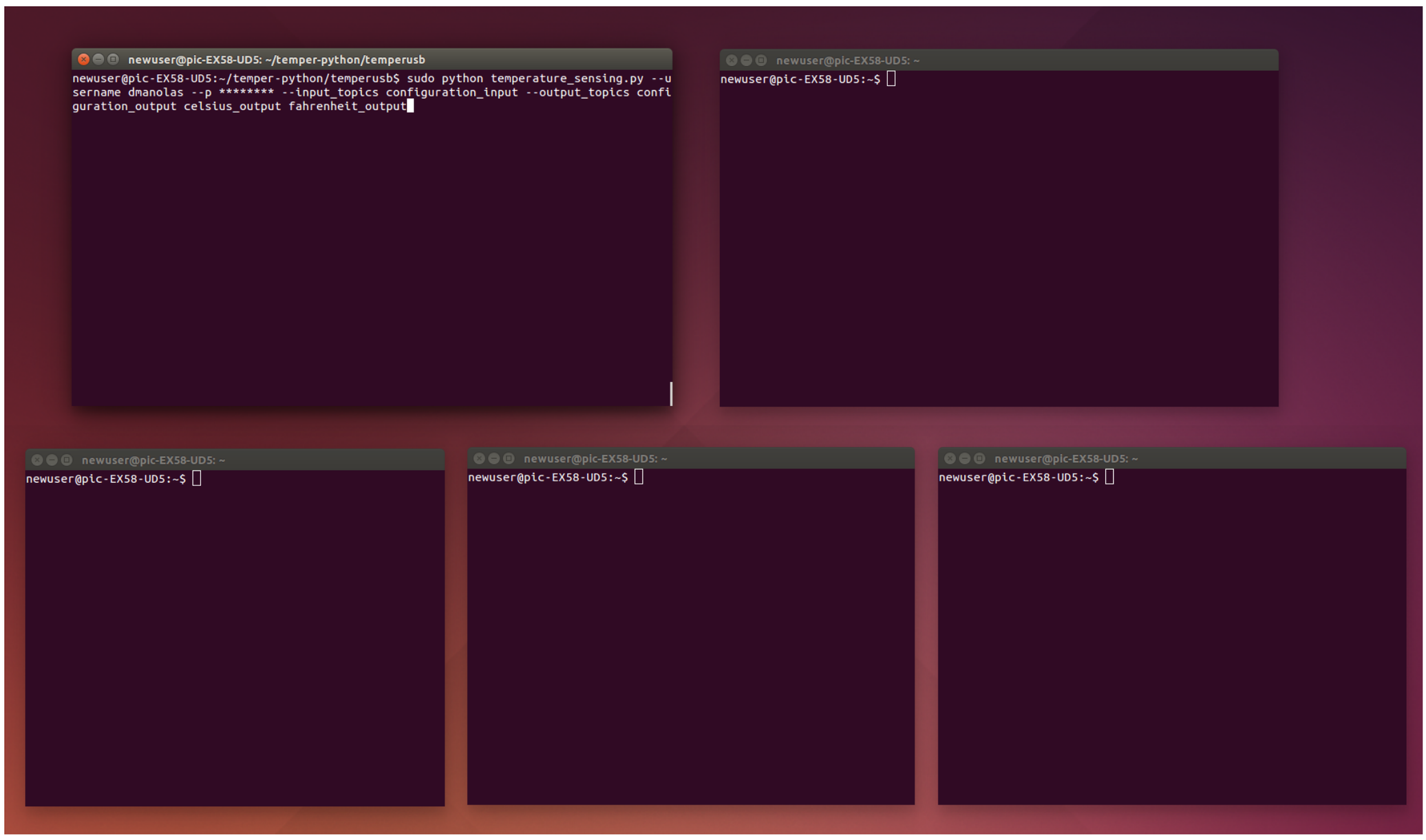
Next up...

We have prepared for you and uploaded the s-type service file necessary for the service to be deployed. Navigate to **synaisthisi-container/ examples/ temper s-type service** file in the repository and download the **temperature_sensing.py** file.

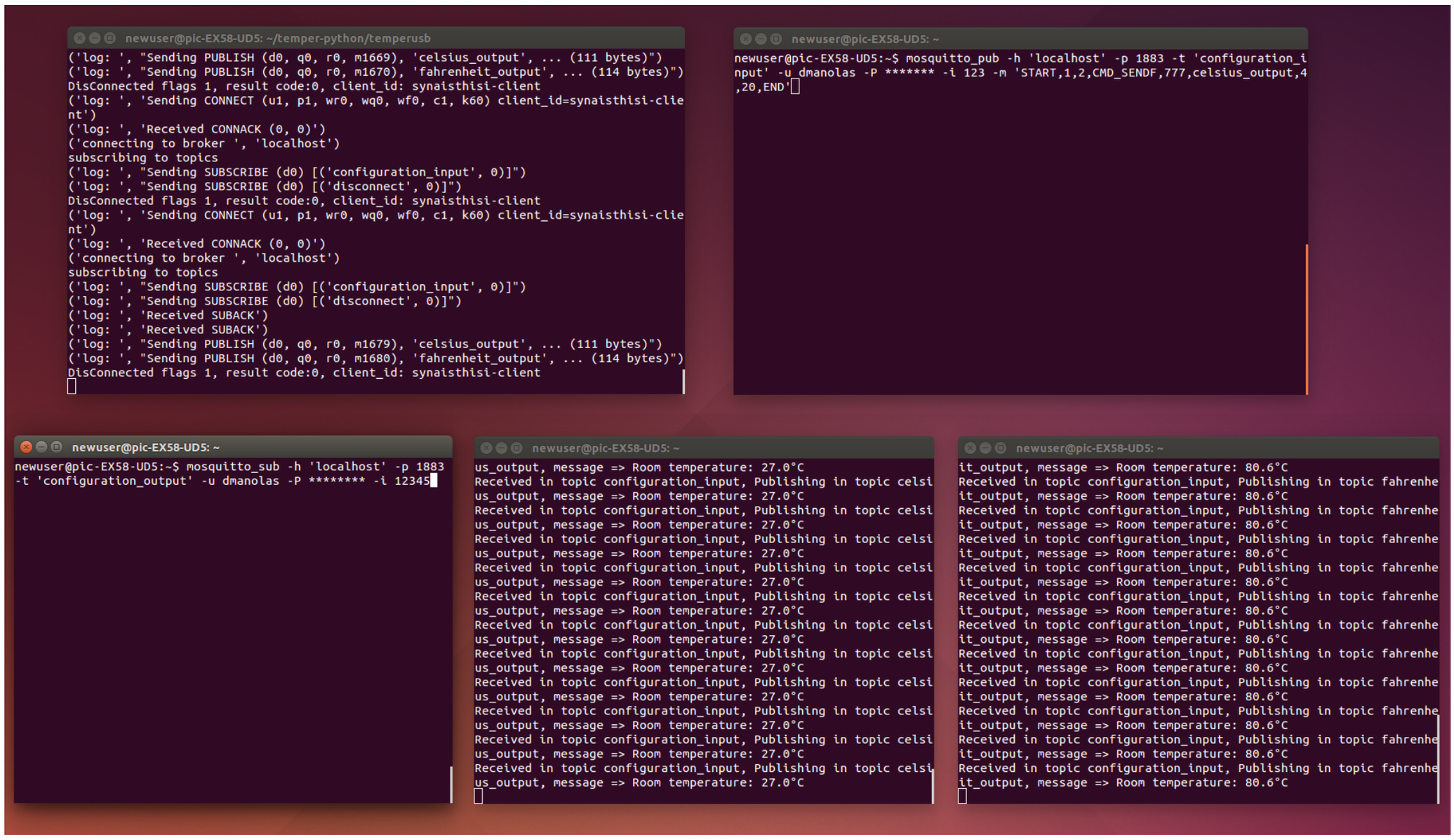
The TEMPer repository is also needed. You can download it here.

> When downloading is done, extract the contents of the file in your *Home* directory and then, place **temperature_sensing.py** inside *temper-python/temperusb*.
>
> *Open a terminal and navigate to this folder using the following command:*
>
> ```
> cd temper-python/temperusb
> ```
>
> *Then, execute the command below to start the service:*
>
> **Tip:** Log in SYNAISTHISI now, if you haven't already!
>
> ```
> sudo python temperature_sensing.py --username <your username> --p <your password> --input_topics configuration_input --outp
> ```

That should be all!

> **If you want to make sure that everything is working, follow these next steps:**
>
> Open another 4 different terminal windows and organize them side by side with the one that's already open in your screen, so that every one of them is visible simultaneously.
>
> *For example, your screen should look something like this:*
>
> 
>
>
> *Type each of the following 4 commands on a different blank terminal:*
>
> ```
> 1. mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m
>
> 2. mosquitto_sub -h 'localhost' -p 1883 -t 'configuration_output' -u <your username> -P <your password> -i 12345
>
> 3. mosquitto_sub -h 'localhost' -p 1883 -t 'celsius_output' -u <your username> -P <your password> -i 456
> ```
>
> **Hint:** Where you execute the above command will dictate where the temperature output is going to show in **Celsius** degrees!
>
> ```
> 4. mosquitto_sub -h 'localhost' -p 1883 -t 'fahrenheit_output' -u <your username> -P <your password> -i 789
> ```
>
> **Hint:** Where you execute the above command will dictate where the temperature output is going to show in **Fahrenheit** degrees!

A few moments in the process your screen should start looking like this:



Notice how, the terminals in the bottom right of the screenshot, display the room temperature in Celsius/Fahrenheit and refresh the output stream every few seconds.

Well, the *"seconds"* part is actually configurable...

> Changing the parameters in the end of command no.1 of the listed commands can affect the information you are getting in various ways.
>
> *For example, in...*
>
> ```
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,
> ```
>
> - The '**4**' at the end stands for the refresh rate (in seconds) of the output stream.
> - The '**20**' refers to the duration of the selected refresh rate, after witch it returns to its default refresh rate of 1 sec.
> - Instead of "**celsius_output**" you can try "**fahrenheit_output**" to change the Fahrenheit output stream refresh rate.
>
> Here are some other commands to experiment with on your own and get a little more acquainted with the whole process.
>
> ```
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,CMD_SEN
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,CMD_SEN
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,CMD_CLE
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,CMD_STA
> mosquitto_pub -h 'localhost' -p 1883 -t 'configuration_input' -u <your username> -P <your password> -i 123 -m 'START,1,2,CMD_EXI
> ```

You can now proceed to our next wiki, Deploying an SPA Micro-service for face detection.

Updated 23 hours ago