Instructions

To run the IoT temperature monitoring system using MQTT on Jupyter Notebook, follow these instructions:

<u>First,</u>

Use these instructions to install the Mosquitto MQTT broker locally:

<u>On Windows:</u>

1.Get the Mosquitto app:

Visit the Downloads page for Mosquitto.

Get the Windows installation here.

2. Set up Mosquitto:

Launch the installation that you downloaded.

As directed by the installation wizard, follow the steps.

Select the components (such as Mosquitto Broker, Mosquitto Clients, etc.) that you wish to install.

Finish the installation process after modifying the installation directory.

Launch the Broker Mosquitto:

Mosquitto is typically configured to run as a service after installation.

You can use Windows' Services Manager to start and stop the Mosquitto service.

Alternatively, to start the service, open a command prompt and type: net start mosquitto.

2. Confirm Installation:

Run the command "mosquitto -h" at the command prompt. If Mosquitto is installed correctly, it should show its help information.

On macOS:

1. Use Homebrew to install Mosquitto:

Launch the terminal.

To install Mosquitto using Homebrew, type brew install mosquitto.

2. Launch the Broker Mosquitto:

Once installed, Mosquitto ought to launch on its own.

If not, launch the Mosquitto service in Terminal by running brew services start mosquitto.

3. Examine the Installation:

Run mosquitto -h after opening Terminal. It should show the help page for Mosquitto, indicating that the installation was successful.

To install Mosquitto on Linux (Ubuntu/Debian), open Terminal:

For the package list to be updated, run sudo apt update.

To install Mosquitto and the command-line clients, use sudo apt install mosquitto mosquitto-clients.

Launch the Broker Mosquitto:

The moment Mosquitto is installed, it ought to launch automatically.

If not, use sudo systemctl start mosquitto to launch the service.

Examine the Installation:

Run mosquitto -h after opening Terminal. It should show the help page for Mosquitto, indicating that the installation was successful.

Verification: Using the MQTT client libraries in your programming language or Mosquitto's command-line clients (mosquitto_pub, mosquitto_sub), you can test the broker by subscribing and publishing to test topics after installation.

You should be able to install and launch the Mosquitto MQTT broker locally on your operating system by following these steps. Depending on how your system is configured specifically, adjustments could be required.

Second,

Here is a more thorough how-to instruction for using an IPython or Jupyter Notebook to execute the MQTT simulation code for temperature monitoring:

Required conditions:

1. Python Workspace:

Make sure Python is installed on your computer.

2. Jupyter Notebook or IPython:

If you haven't previously, install IPython or Jupyter Notebook in your Python environment.

3. Libraries Required:

In case you haven't already, install the paho-mqtt library. You can run to do this!paho-mqtt in a laptop cell using pip install.

How to Execute the Code:

1. Open Jupyter Notebook: Open the environment for Jupyter Notebook.

2. Create a New Notebook or Open an Existing One: To execute the MQTT simulation code, create a new notebook or open an existing one.

3. Copy the code, then paste it:

Copy and paste the complete MQTT simulation code (already provided) into a new notebook cell.

4. Run the Code Cell: You may either click the "Run" button in the notebook toolbar or click within the code cell and press Shift + Enter to start it.

5. Execution Flow: When the code runs, it will carry out the following operations:

Establish a connection with a local MQTT broker using localhost and port 1883 by default.

Three temperature sensors are simulated.

generating random temperature readings between 10°C and 40°C.

In the notebook's output area, display a header that reads "IoT Temperature Monitoring" along with subsections for every sensor.

sensors1,2, and 3 are producing erroneous temperature readings between 10°C and 40°C.

In the notebook's output area, display a header that reads "IoT Temperature Monitoring" along with subsections for every sensor.

6. Communicate Using the Notebook:

Display of Temperature Readings: Watch as each sensor's displayed temperature reading updates in real time on the notebook's output area.

These comprehensive instructions will help you to run the MQTT simulation code for IoT temperature monitoring in an IPython or Jupyter Notebook environment, view the simulated sensor readings, and control the notebook's output to keep an eye on the temperature.