UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

SNAP4CITY | KM4CITY

Data Ingestion Tutorial

Corso: Sistemi Distribuiti, UNIFI, DISIT lab

Data Ingestion Step

- **HOW TO Create an IOT Device Model:** https://www.snap4city.org/591
- **HOW TO: Create an IOT Device Instance**: https://www.snap4city.org/590

  [Entity/IoT Directory tool - IotAPP]
- **HOW TO Develop an IOT Application for Data Ingestion** https://www.snap4city.org/593
- **HOW TO Managing Notifications on IOT Application** https://www.snap4city.org/142

Data Ingestion Example

Open Weather data, **Open Pollution data** and **Open Sea Condition data** for each pilot.

Data Source	Data Source Main Address	Periodicity for Data ingestion	Variables
Open Weather	https://openweathermap.org/api	30 mins	• airHumidity • airTemperature • cloudCoverPerc • feelsLike • groundLevel • maxTemperature • minTemperature • pressure • seaLevel • sunrise • sunset • visibility • windDirection • windSpeed

Data Ingestion Example

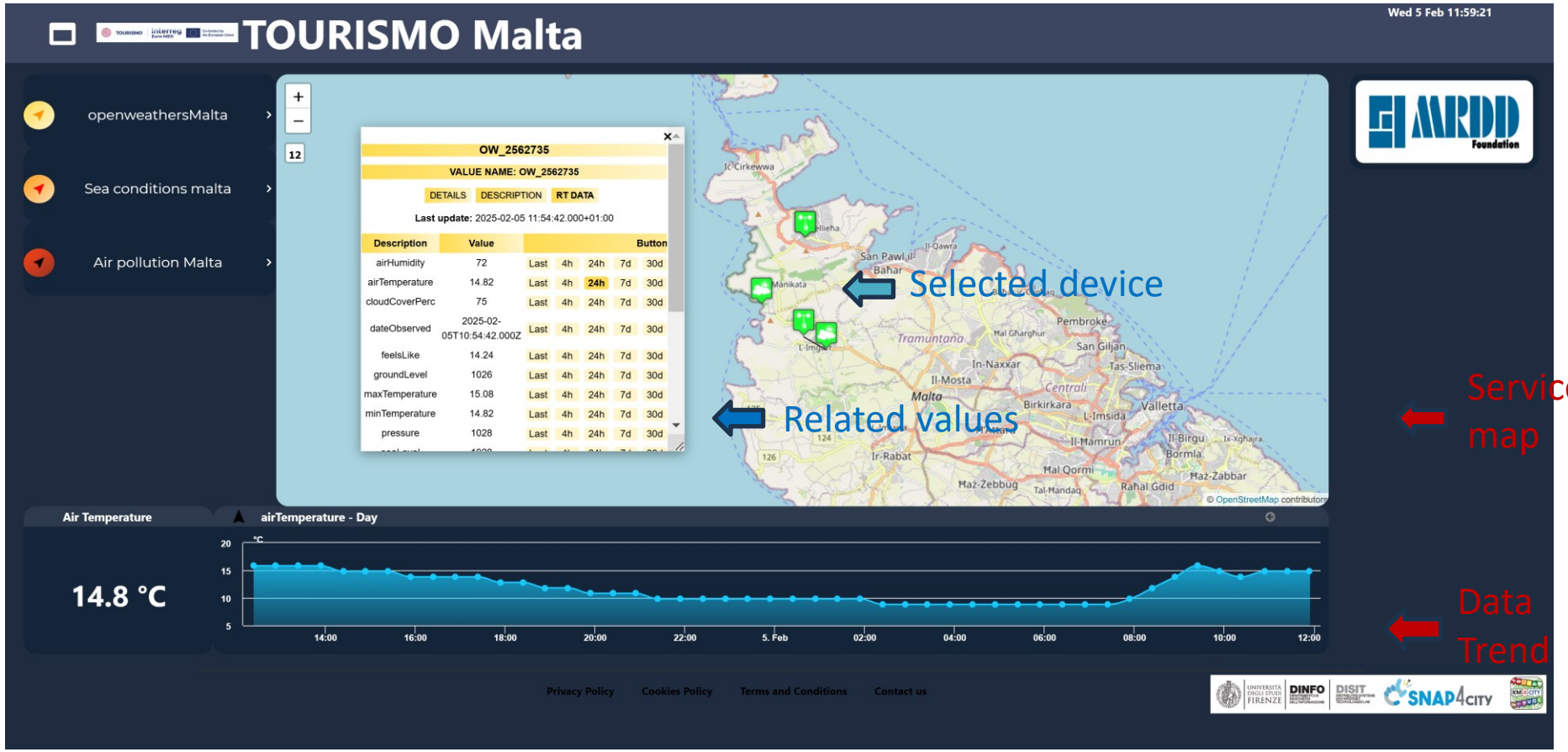
Data Source	Data Source Main Address	Periodicity for Data ingestion	Variables
Sea Conditions	https://open-meteo.com/	60 mins	• oceanCurrentDirection • oceanCurrentVelocity • swellWaveDirection • swellWaveHeight • swellWavePeakPeriod • swellWavePeriod • waveDirection • waveHeight • wavePeriod • windWaveDirection • windWaveHeight • windWavePeakPeriod • windWavePeriod

Data Ingestion Example

Data Source	Data Source Main Address	Periodicity for Data ingestion	Variables
Open Pollution	https://openweathermap.org/api/air-pollution	30 mins	• CO • NO • NO2 • NH3 • O3 • SO2 • PM2.5 • PM10

Dashboard:
https://www.snap4city.org/dashboardSmartCity/view/newTheme.php?iddasboard=NDMzNQ==

Data Visualization in a Dashboard



Devices selector

Selected device

Related values

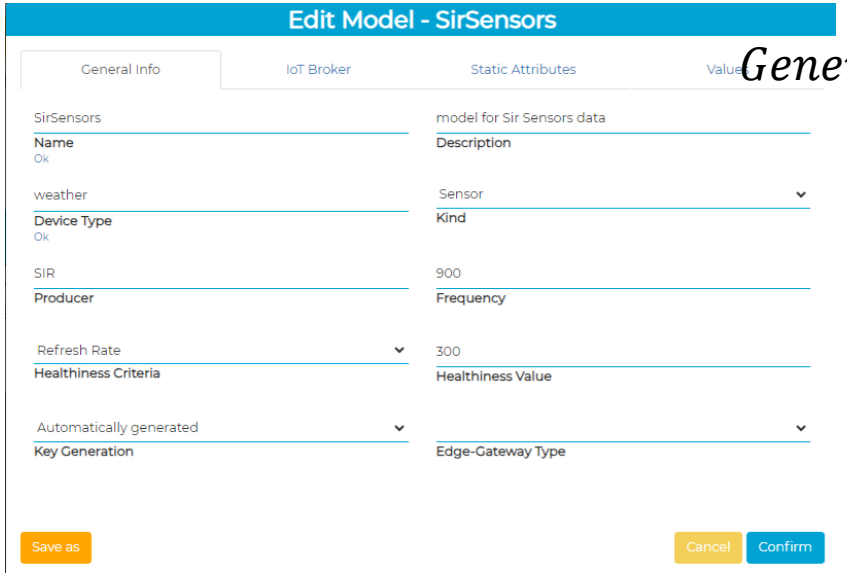
Partner link

Service map

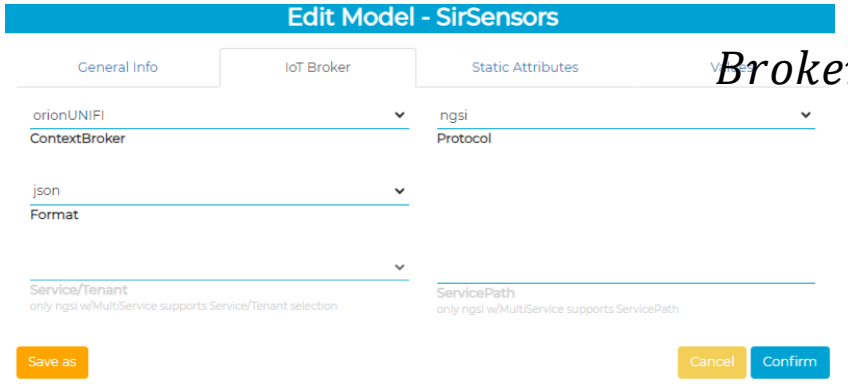
Data Trend

HOW TO Create an IOT Device Model

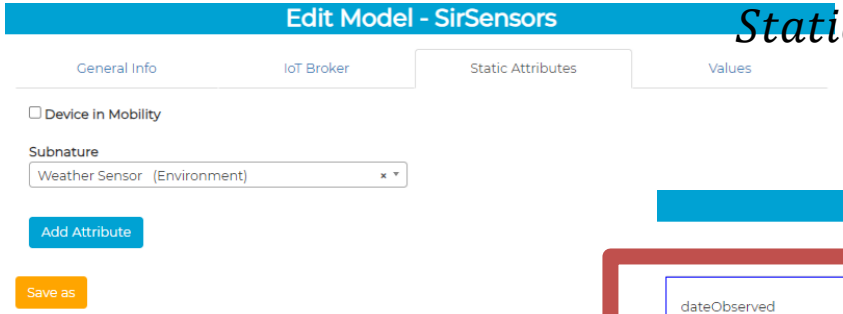
General Info



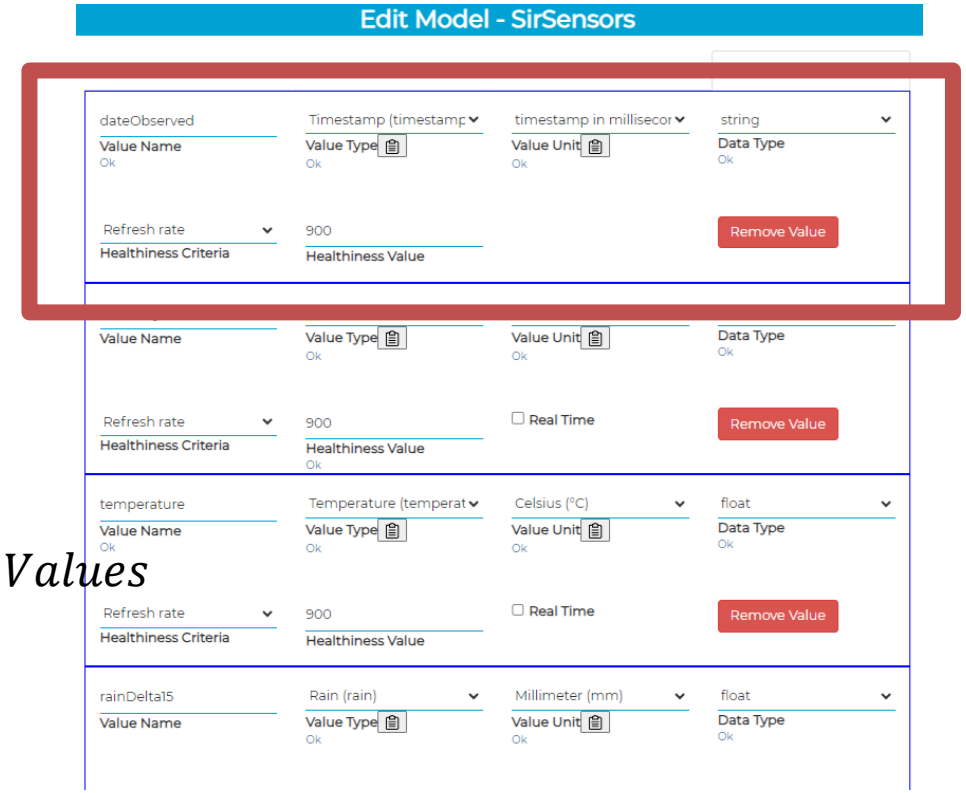
Broker



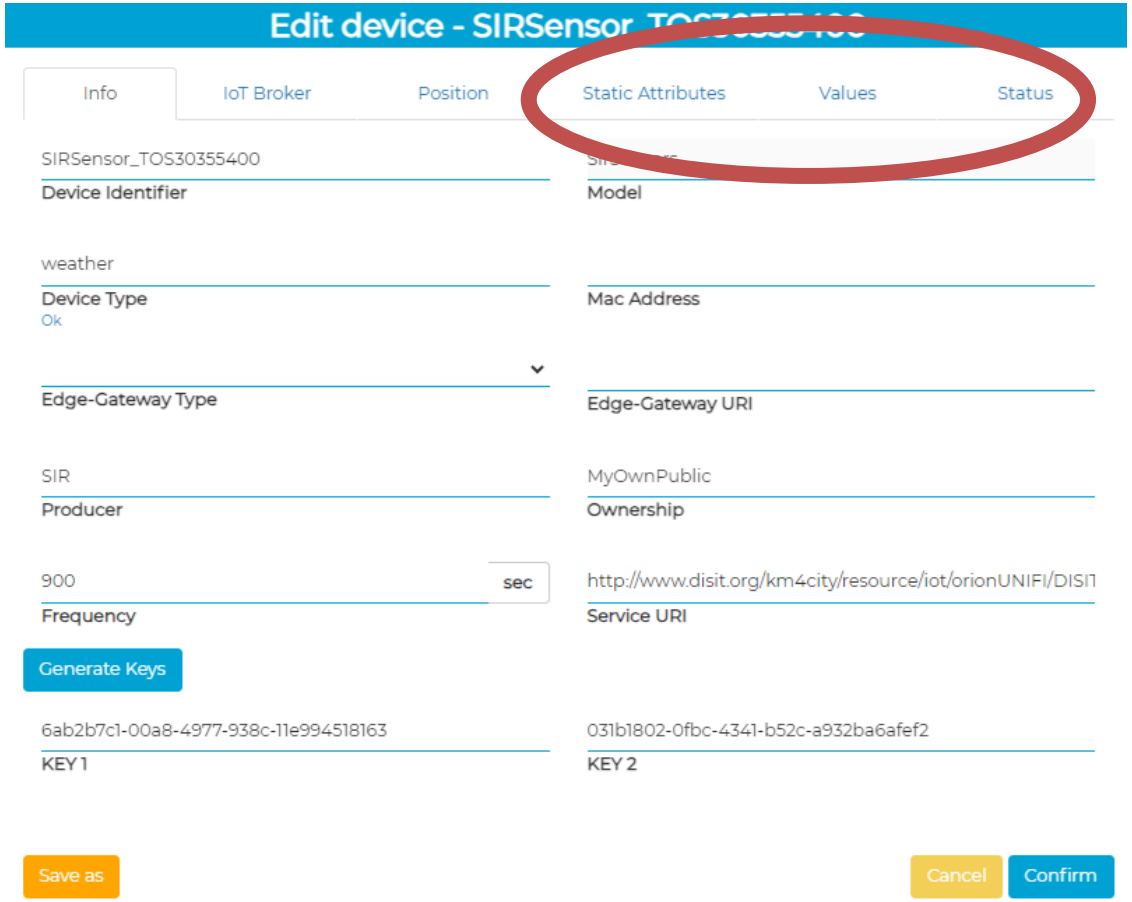
Static Attributes

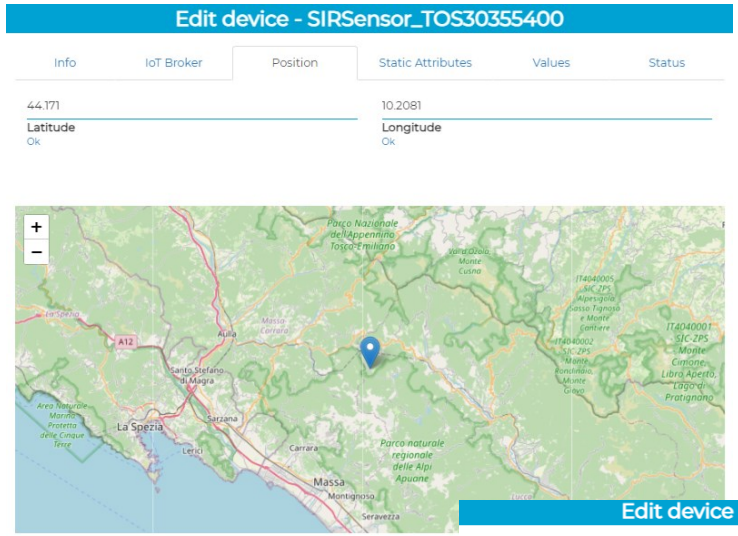


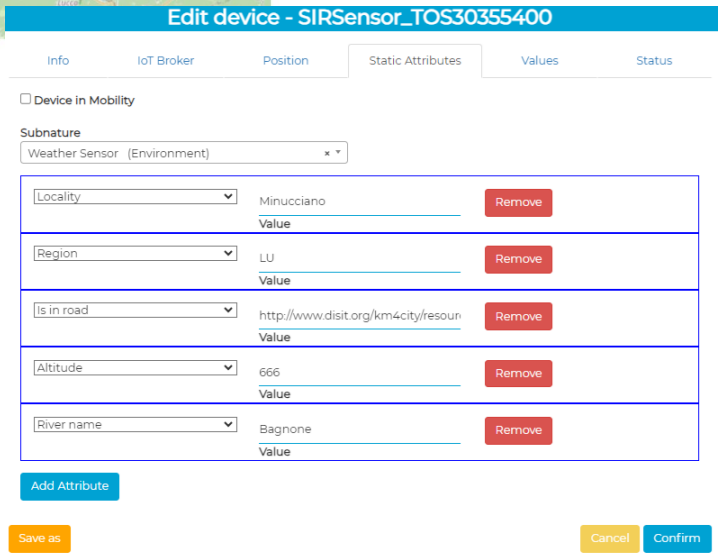
Values



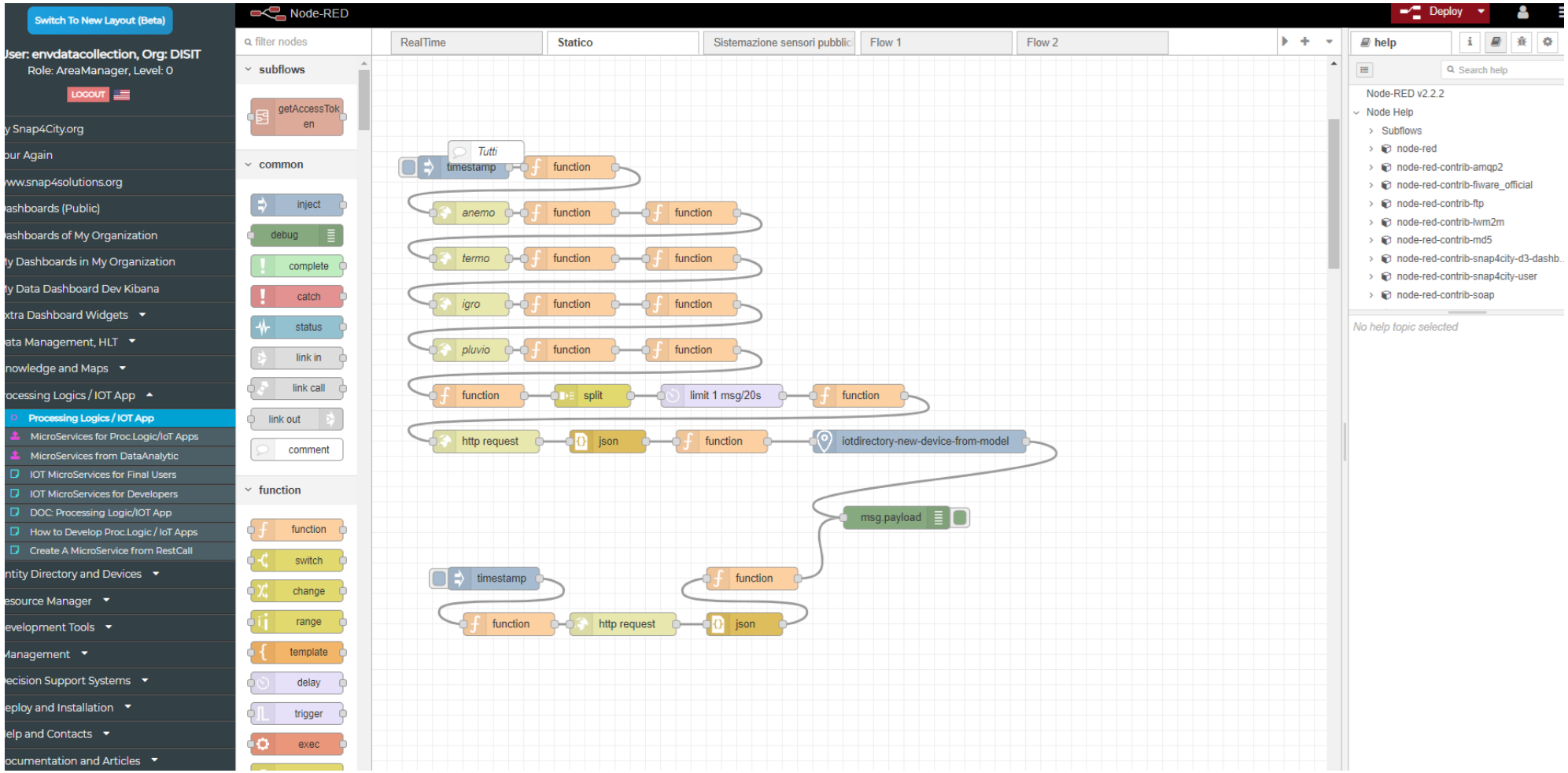
HOW TO Create an IOT Device instance



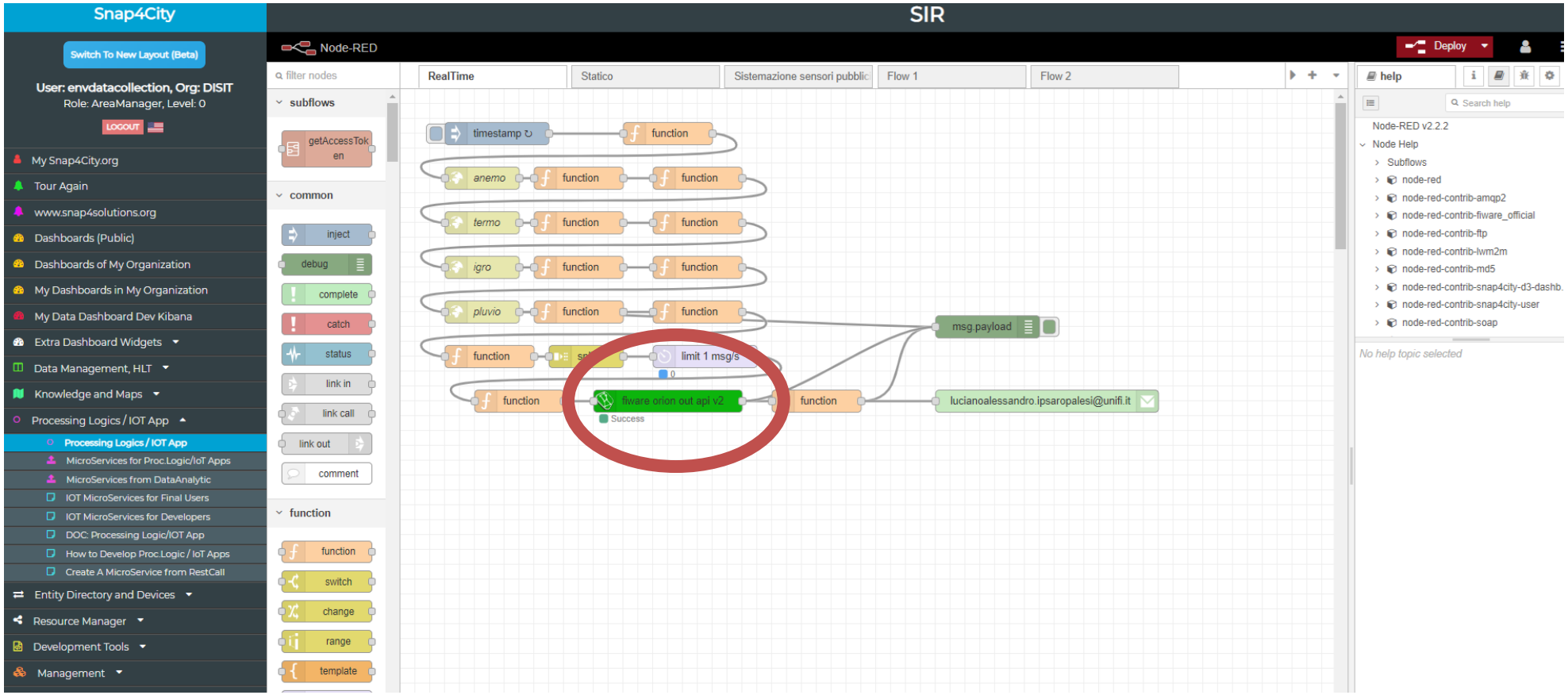




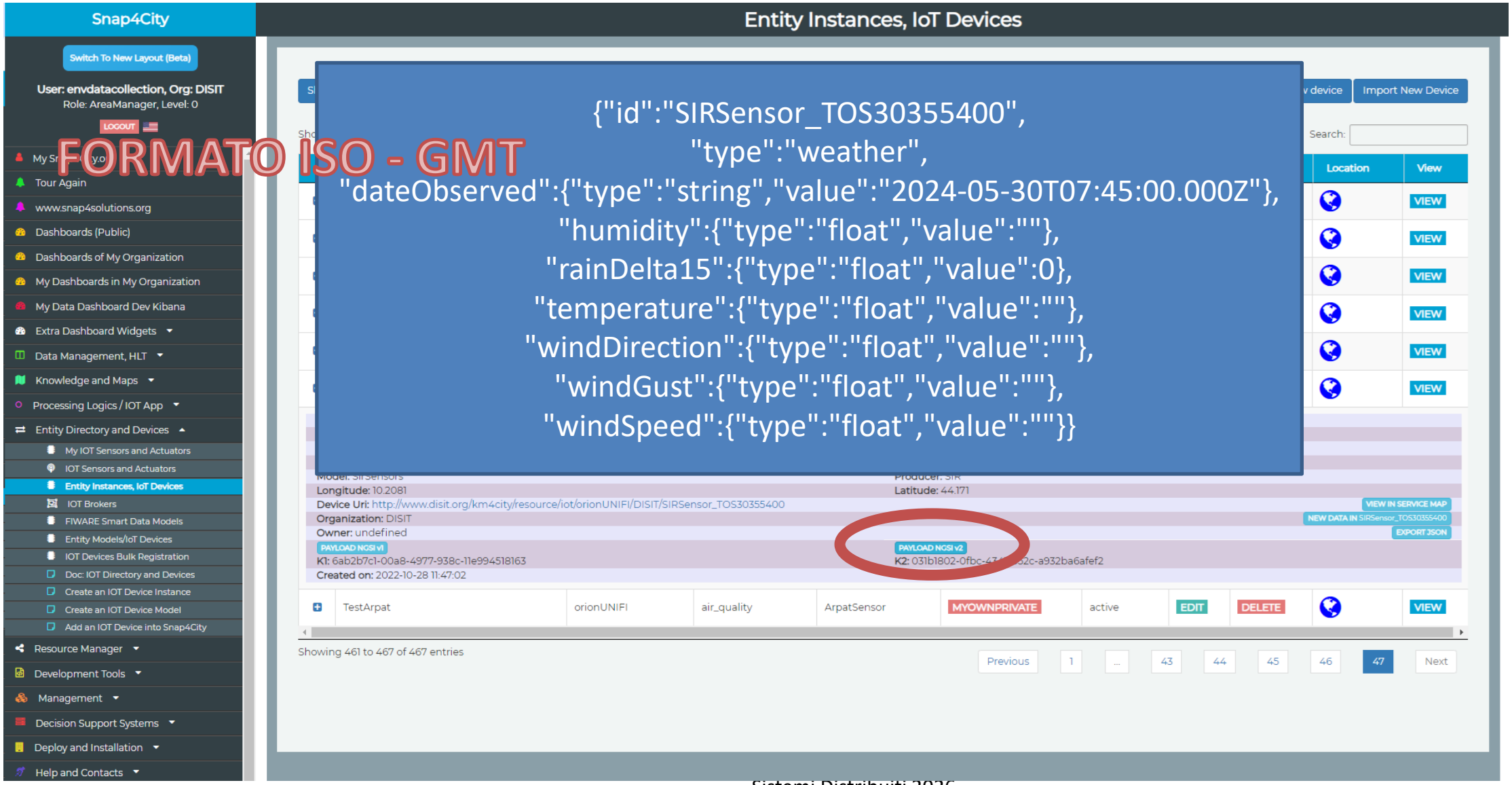
HOW TO Create an IOT Device instance



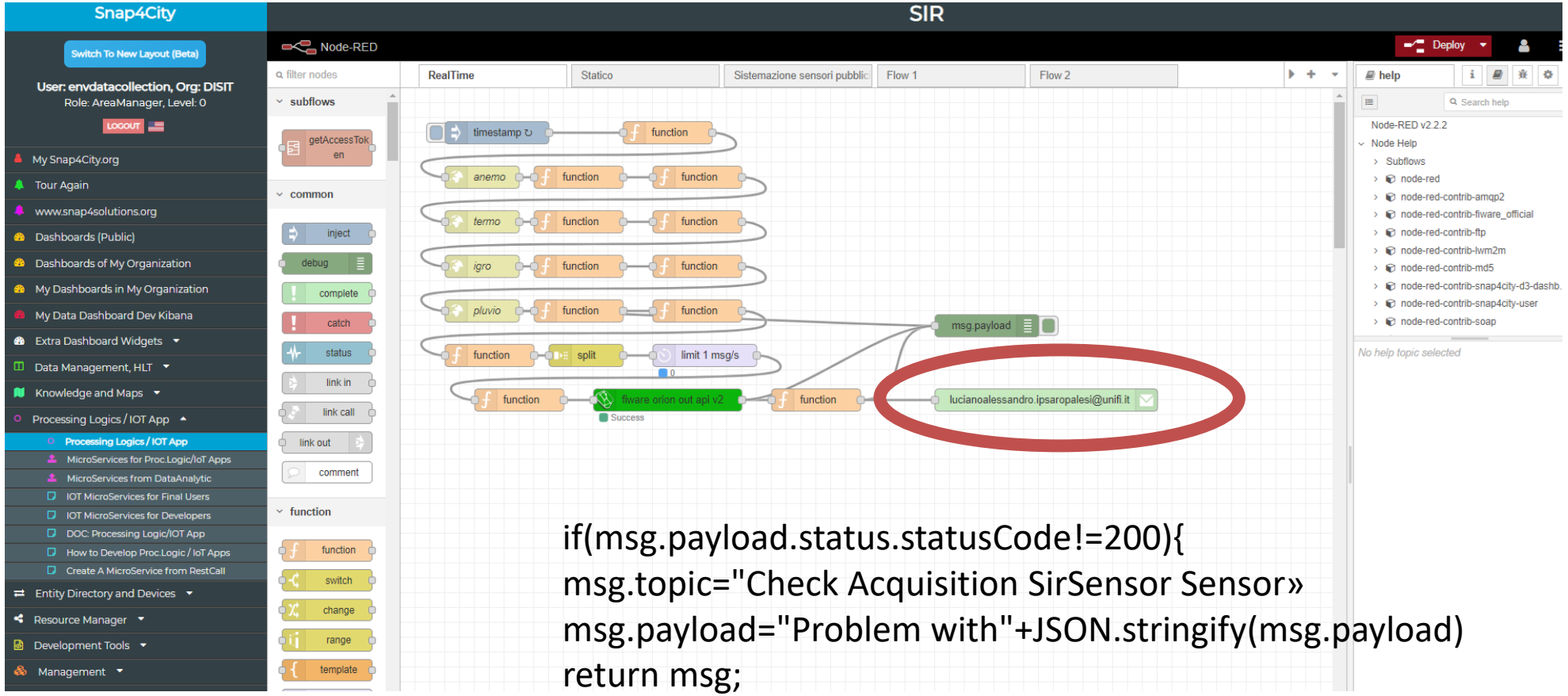
HOW TO Create an IOT Application for Data Ingestion



HOW TO Create an IOT Application for Data Ingestion

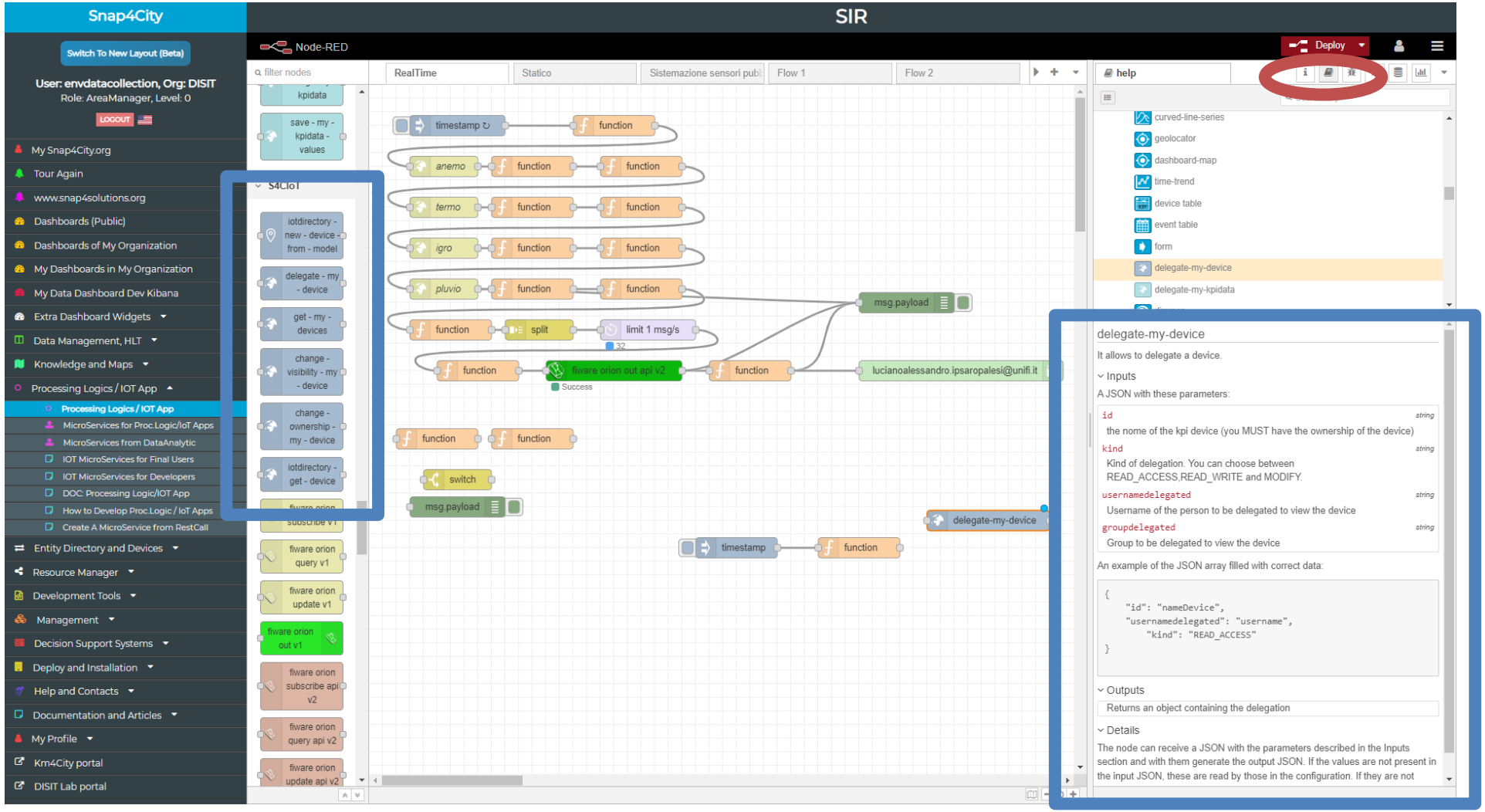


HOW TO Managing Notifications on IOT Application




```
if(msg.payload.status.statusCode!=200){
msg.topic="Check Acquisition SirSensor Sensor»
msg.payload="Problem with"+JSON.stringify(msg.payload)
return msg;
}
return null;
```

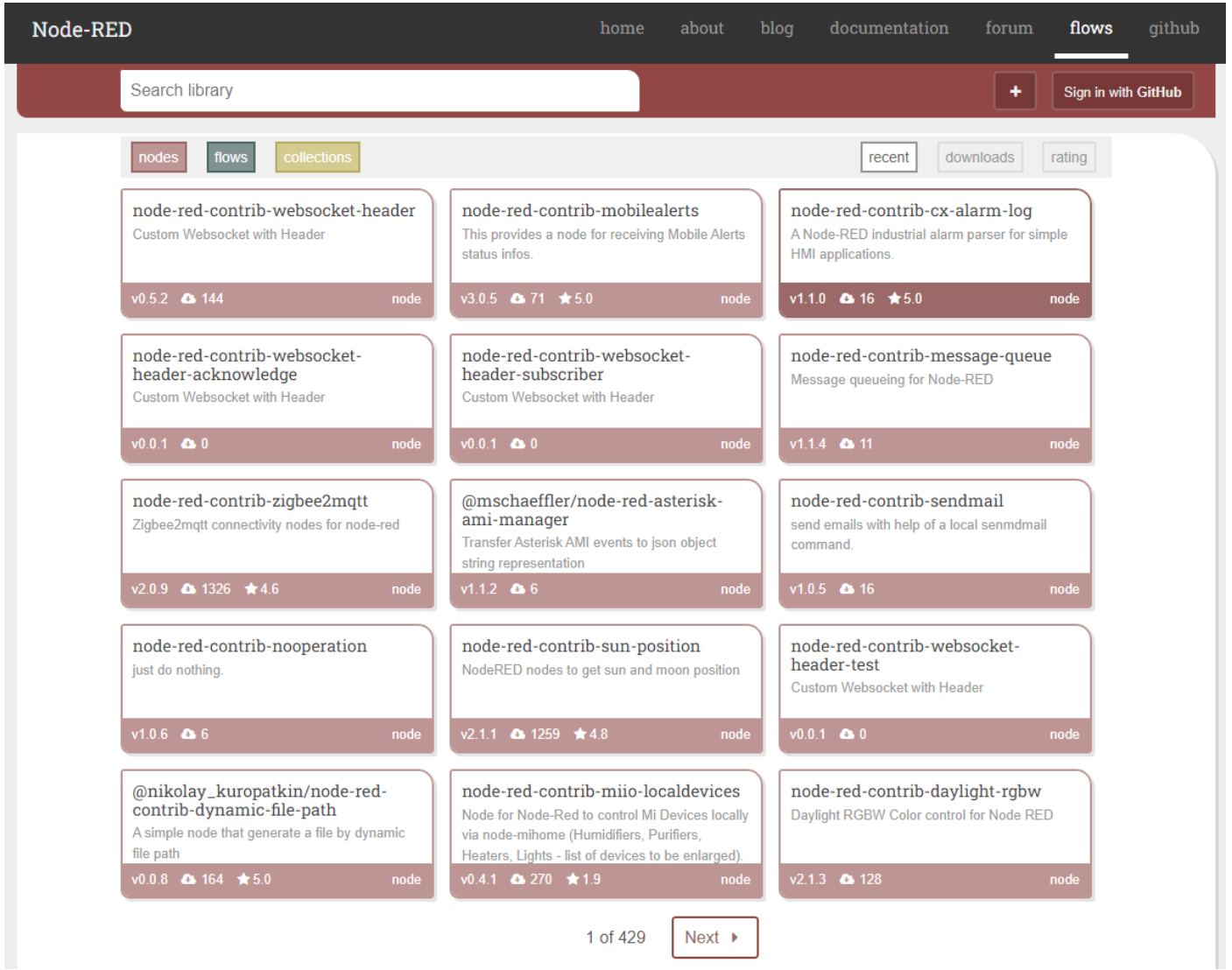
IotApp


Snap4City
SIR
Node-RED
Deploy
Switch To New Layout (Beta)
User: envdatacollection, Org: DISIT
Role: AreaManager, Level: 0
LOGOUT
My Snap4City.org
Tour Again
www.snap4solutions.org
Dashboards (Public)
Dashboards of My Organization
My Dashboards in My Organization
My Data Dashboard Dev Kibana
Extra Dashboard Widgets
Data Management, HLT
Knowledge and Maps
Processing Logics / IOT App
Processing Logics / IOT App
MicroServices for Proc.Logic/IoT Apps
MicroServices from DataAnalytic
IOT MicroServices for Final Users
IOT MicroServices for Developers
DOC: Processing Logic/IOT App
How to Develop Proc.Logic / IoT Apps
Create A MicroService from RestCall
Entity Directory and Devices
Resource Manager
Development Tools
Management
Decision Support Systems
Deploy and Installation
Help and Contacts
Documentation and Articles
My Profile
Km4City portal
DISIT Lab portal
RealTime
Statico
Sistemazione sensori publ
Flow 1
Flow 2
help
delegate-my-device
It allows to delegate a device.
Inputs
A JSON with these parameters:
id
the nome of the kpi device (you MUST have the ownership of the device)
kind
Kind of delegation. You can choose between READ_ACCESS,READ_WRITE and MODIFY.
usernamedelegated
Username of the person to be delegated to view the device
groupdelegated
Group to be delegated to view the device
An example of the JSON array filled with correct data:
Outputs
Returns an object containing the delegation
Details
The node can receive a JSON with the parameters described in the Inputs section and with them generate the output JSON. If the values are not present in the input JSON, these are read by those in the configuration. If they are not

Node-RED Libraries

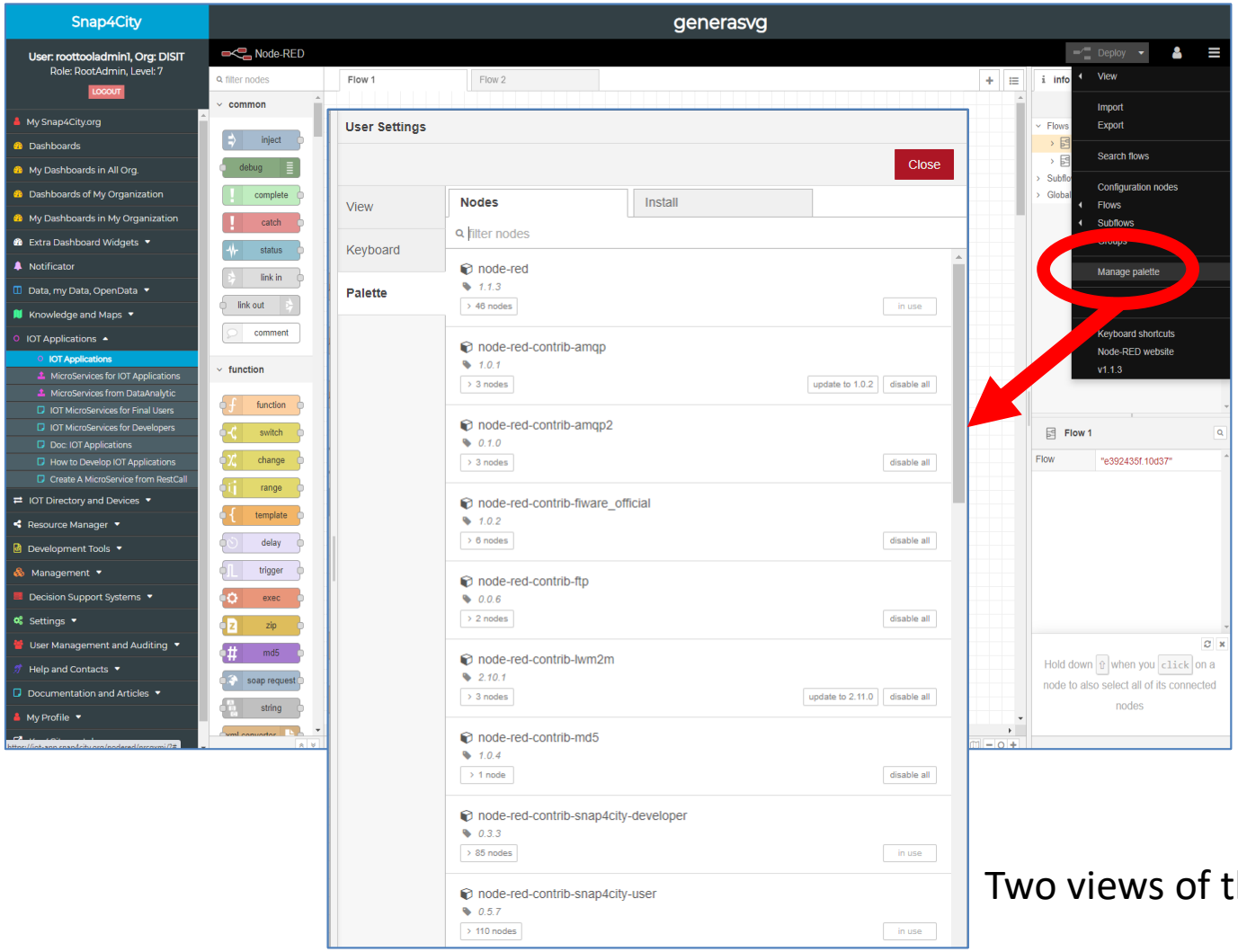
Node-RED

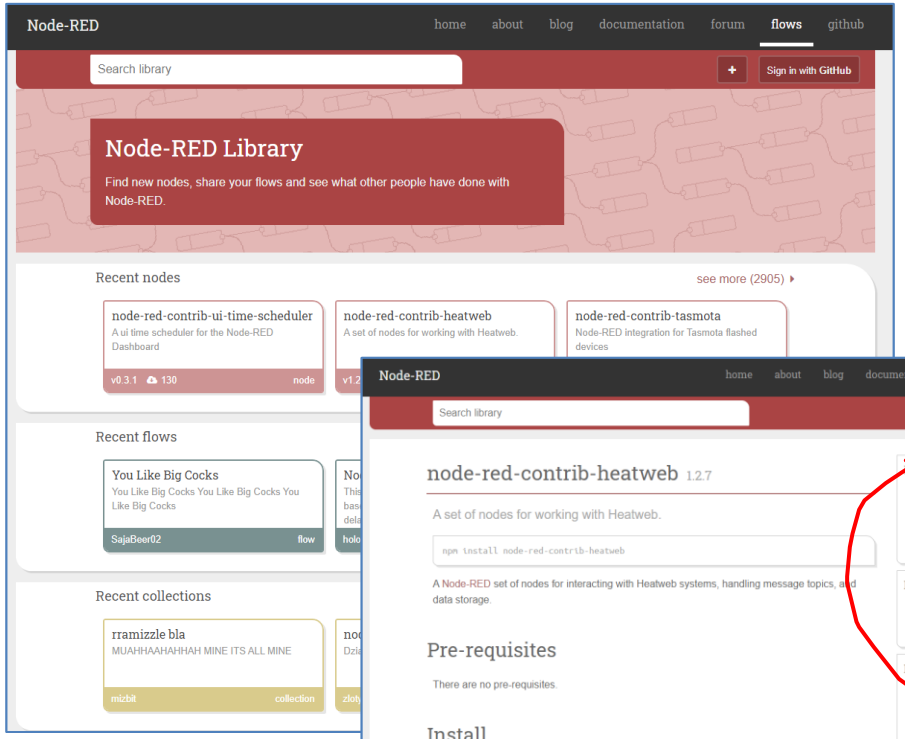
https://flows.nodered.org/search?term=

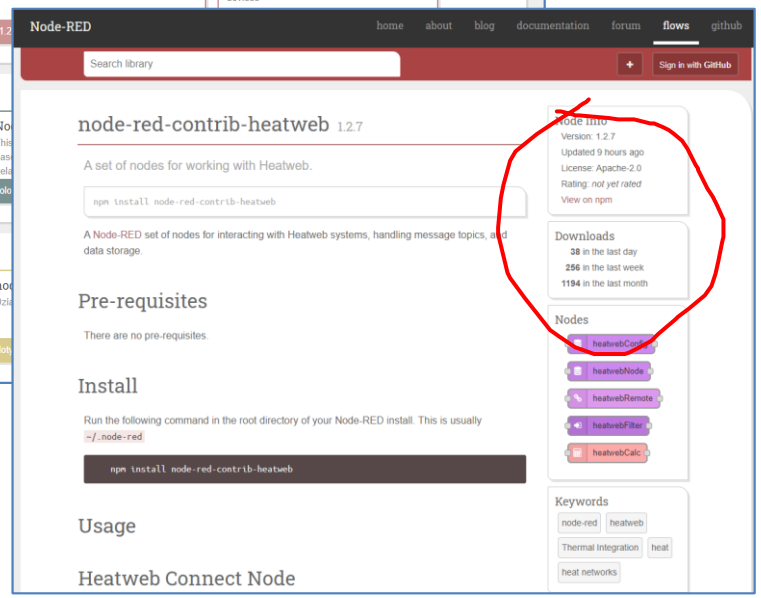


Load Library from Palette

https://flows.nodered.org/





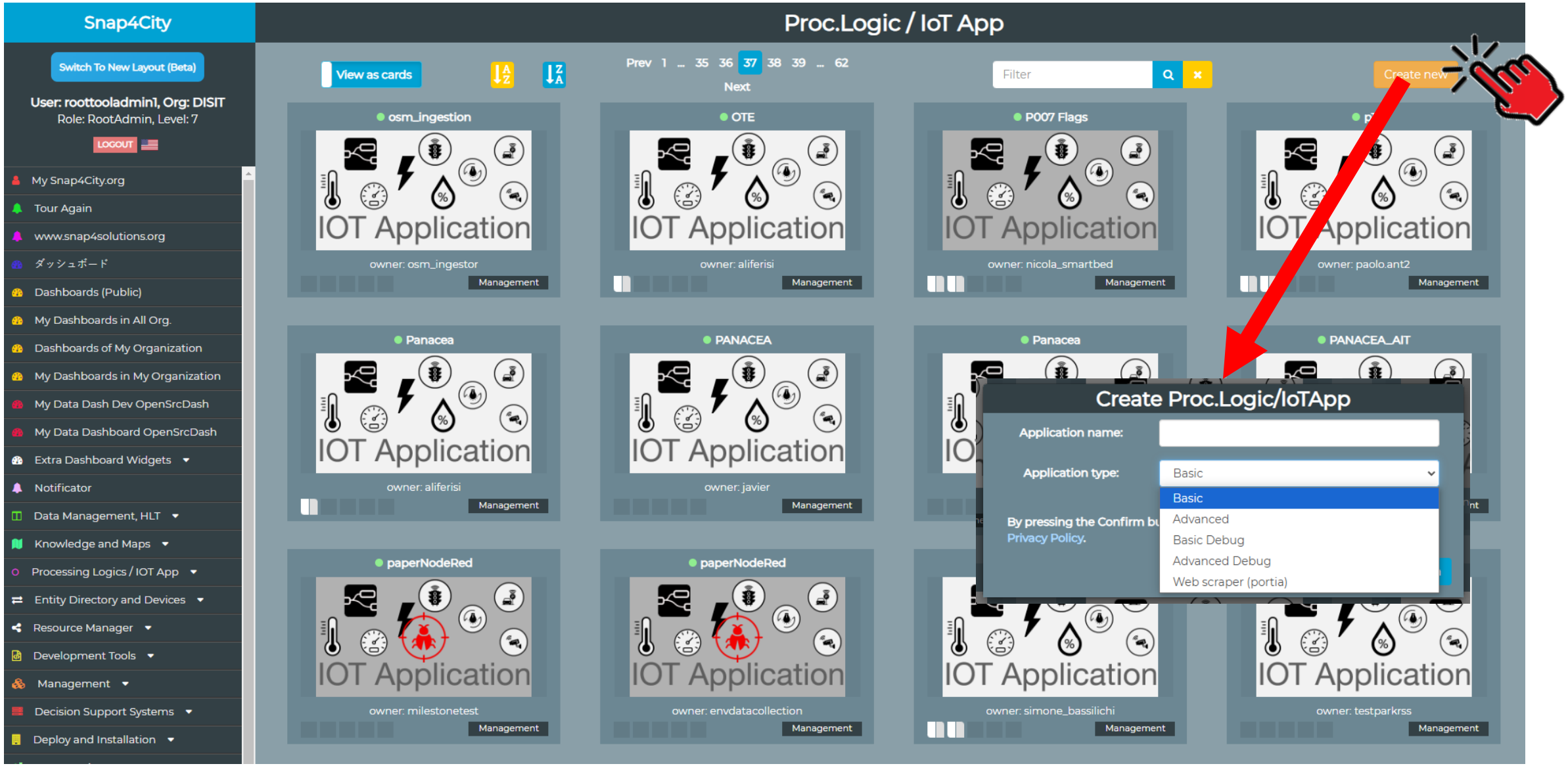


Two views of the same libraries

Node-RED in SnaP4City

How to Design

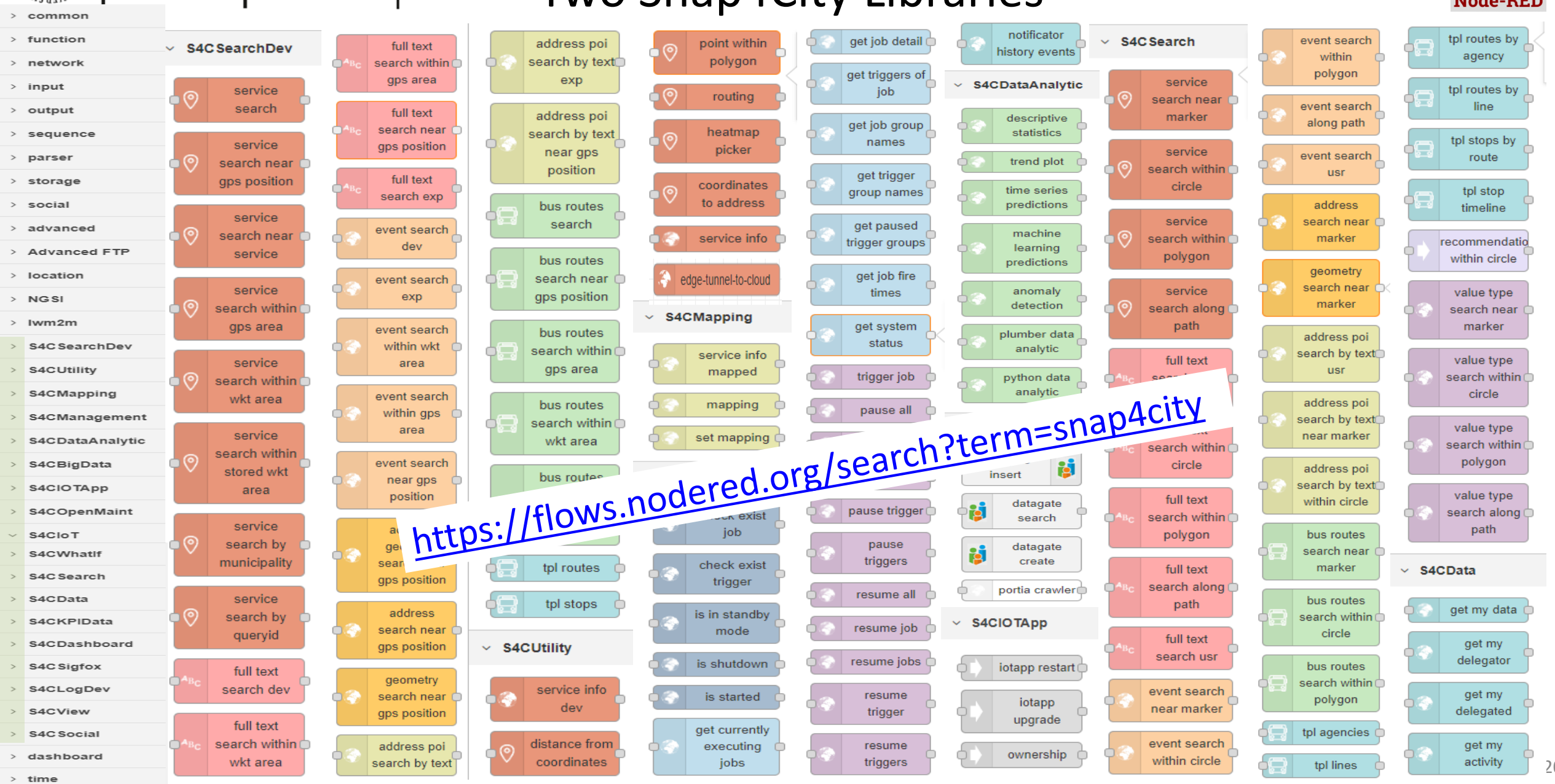
1. **Business Logic** is going to be implemented in Proc.Logic (IoT App), with a set of flows.
2. **Decompose your problem** and sequence diagram in single Data/event Flows, from client side and server side.
3. **Identify the single Data/Event Flow**, as those that start from a certain event (periodic or provoked from other messages), and that finish with: sending of data in the storage, change status, send an event, provide a message into a dashboard, send an email, etc.
4. **Design the single Data/Event Flows** with a mixt of possible **activities**.
    1. The design can be performed using data flow diagrams.
    2. It can have sequences, switch, serialization, packing, joining, distribution, communication, transformation, search, etc.
5. When the design of Data/Event Flow mechanism is clear the designers can pass to directly sketch the flow in Node-RED which is a visual programming.
6. **Incrementally improve the Proc.Logic** (IoT App) Node-RED flows by adding nodes needed
7. **Once obtained the Proc.Logic** (IoT App) Node-RED flows in the correct data model you can send data to the ingestion broker, but also perform many other actions on several services.


Snap4City
Proc.Logic / IoT App
Switch To New Layout (Beta)
User: roottooladmin1, Org: DISIT
Role: RootAdmin, Level: 7
LOGOUT
My Snap4City.org
Tour Again
www.snap4solutions.org
ダッシュボード
Dashboards (Public)
My Dashboards in All Org.
Dashboards of My Organization
My Dashboards in My Organization
My Data Dash Dev OpenSrcDash
My Data Dashboard OpenSrcDash
Extra Dashboard Widgets
Notificator
Data Management, HLT
Knowledge and Maps
Processing Logics / IOT App
Entity Directory and Devices
Resource Manager
Development Tools
Management
Decision Support Systems
Deploy and Installation
View as cards
Prev 1 ... 35 36 37 38 39 ... 62
Next
Filter
Create new
osm_ingestion
owner: osm_ingestor
OTE
owner: aliferisi
P007 Flags
owner: nicola_smartbed
owner: paolo.ant2
Panacea
owner: aliferisi
PANACEA
owner: javier
Panacea
PANACEA_AIT
paperNodeRed
owner: milestonetest
paperNodeRed
owner: envdatacollection
owner: simone_bassilichi
owner: testparkrss
IOT Application
Management
Create Proc.Logic/IoTApp
Application name:
Application type:
Basic
Advanced
Basic Debug
Advanced Debug
Web scraper (portia)
By pressing the Confirm bu
Privacy Policy.

UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

Sept 2024 collection
Two Snap4City Libraries

SNAP4CITY | Node-RED

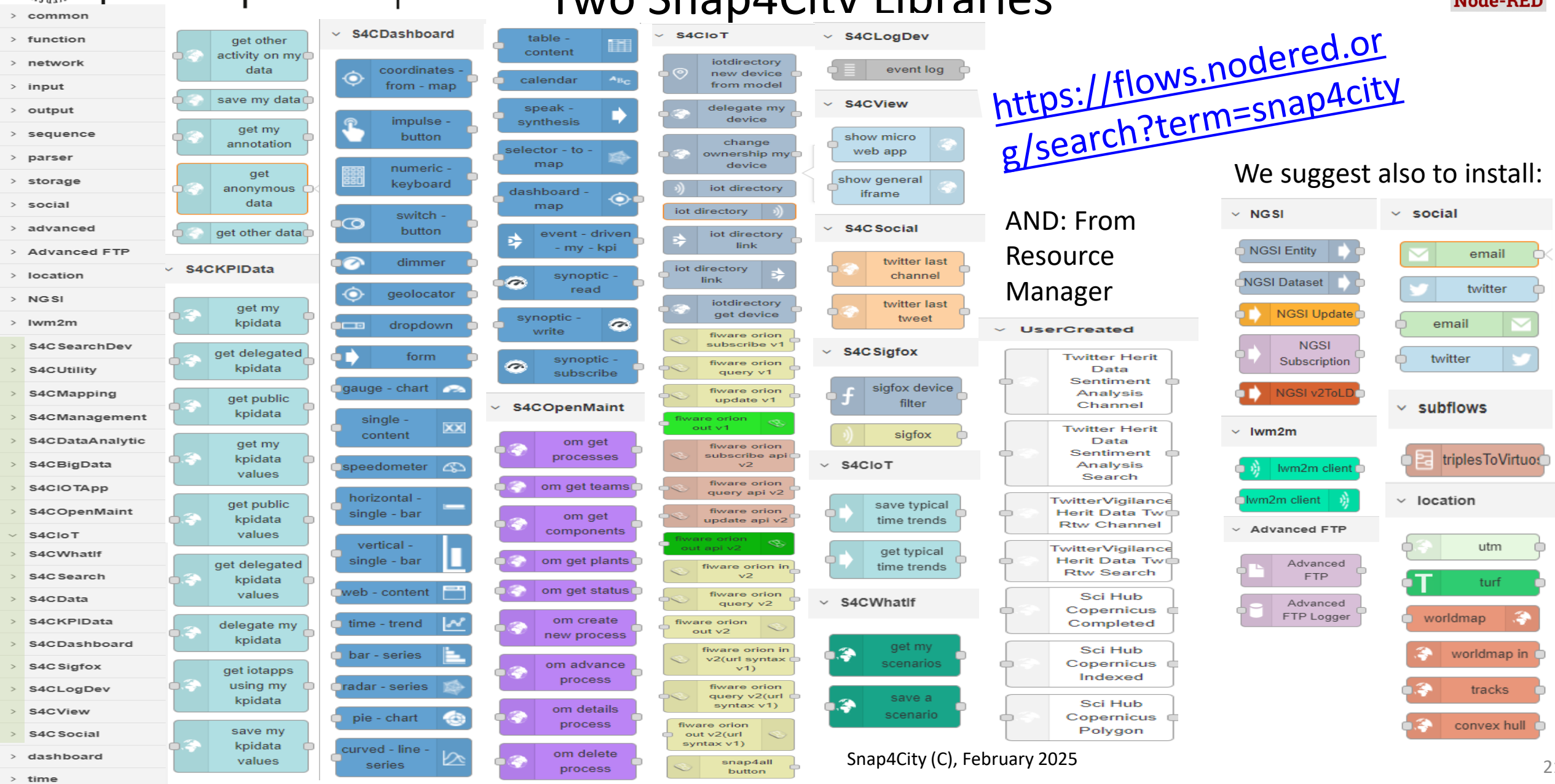


https://flows.nodered.org/search?term=snap4city

Sept 2024 collection
Two Snap4Citv Libraries

https://flows.nodered.org/search?term=snap4city

We suggest also to install:



AND: From Resource Manager

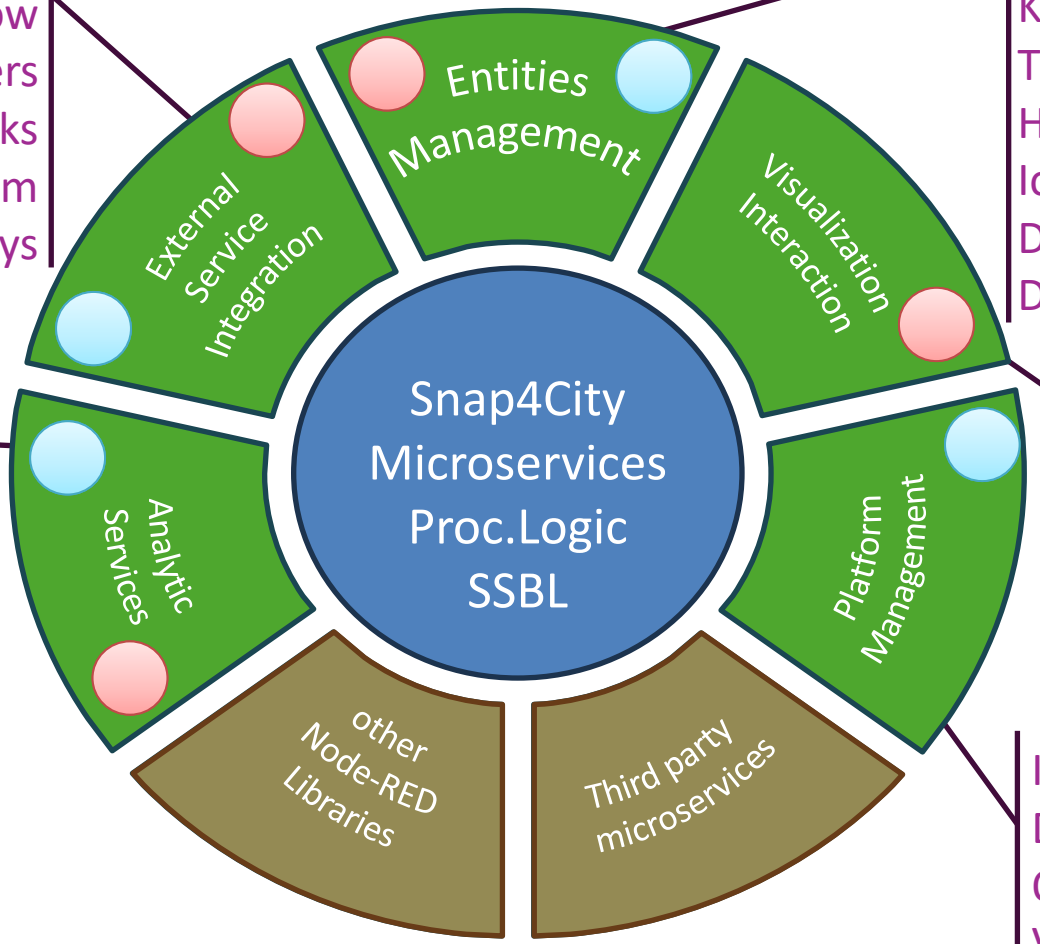
The Processing Logic (IoT App) microservices

Actually, there are more than 180 nodes/blocks in the Snap4City libraries on Processing Logic (IoT App) which can really facilitate your life and save you time in producing Smart Applications for composition of the following microservices and using those that you can install from internet, thousands of functionalities:

- **Data ingestion**: more than 100 protocols IOT and Industry 4.0, web Scraping, external services, any protocol database, etc.
- **Data access**: save/retrieve data, query search on expert system, georeverse solution, search on expert system Km4City ontology, call to Smart City API, etc.
- **Data Transformation/transcoding:** binary, hexadecimal, XML, JSON, String, any format
- **Integration**: CKAN, Web Scraping, FTP, Copernicus satellite, Twitter Vigilance, Workflow OpenMaint, Digital Twin BIM Server, any external service REST Call, etc.
- **Manipulation of complex data**: heatmaps, scenarios, typical time trend, multi series, calendar, maps, etc.
- **Access to Smart City Entities and exploitation of Smart City Services**: transport, parking, POI, KPI, personal data, scenarios, etc.
- **Data Analytic**: managing Python native, calling and scheduling Python/Rstudio containers as snap4city microservices (predictions, anomaly detection, statistics, etc.)
- **User interaction on Dashboard**: get data and message from the user interface, providing messages to the user (form, buttons, switches, animations, selector, maps, etc.), send data to special graphical widgets: D3, Highcharts, etc.
- **Custom Widgets**: SVG, synoptics, animations, dynamic pins on maps, etc
- **Event management**: Telegram, Twitter, Facebook, SMS, WhatsApp, CAP, etc.
- **Special tools as**: routing, georeverse, Twitter Vigilance and sentiment analysis, etc.
- **Hardware Specific Devices**: Raspberry Pi, Android, Philips, video wall management, etc.
- **Etc**. etc.

MicroServices Areas

> 60.000 downloads



Open Data CKAN
Ticket Management, workflow
BIM Servers
Social Networks
Video Management system
Gateways

Data Load / Search / Retrieval
KPI, POI, GIS Data, Scenarios
Time Series, Public transport
High Level Types: heatmaps, ODM,…
IoT / Entity Discovery
Delegation Management
Data Mapping

Data Analytics
Statistic, Optimization
Simulation
Artificial Intelligence
What-if Analysis Support
Geo Utilities Support
Routing & Traffic Flow
MLOps support
Python support
R Studio Support

Dashboards
Widgets: Graphic Libraries
Interactive Widgets
Maps, 3D representations
Synoptics, External Content
Micro Web App

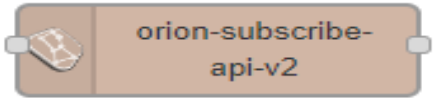
IoTApp Management
Data Logs, A&A, Security
Ownership Management
VPN remote access

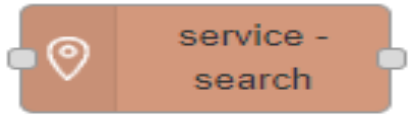
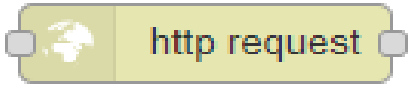
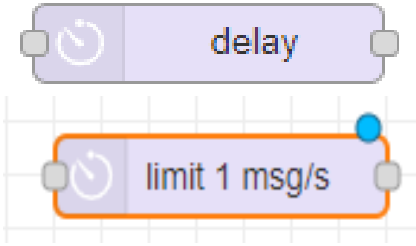
examples

Saving Data on Storage

Even Driven

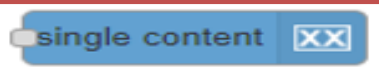
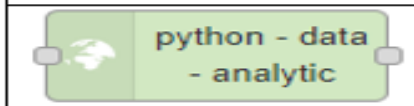
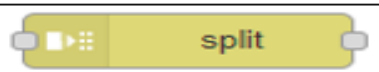
Get Data from Storage

Node shape	Description	Snap4City or standard
	To **generate injection messages** into a flow, scheduled/periodic or on manual demand by click it on left.	standard
	**DATA TRANSFORM** A **JavaScript function**, from a JSON input to one or more JSON outputs, which can be produced by setting it.	standard
	**SAVE to STORAGE via internal BROKER** To send an Entity Message of an Entity Instance into the storage. The Entity Instance has to be registered on Entity Directory (IoT Directory) and you have to be the owner or to be delegated in READ-WRITE to send messages to it. The node represents the broker, so that the same node can be used to send any Entity Message you need. Please manage the error in output.	Snap4city
	**SUBSCRIBE** to an Entity change on BROKER To subscribe the Processing Logic (IoT App) to receive event-driven notifications related to Entity Instances changes. The node is substantially a listener connected to an Orion Broker. You can subscribe to many Entities and then to get all of them from the output of the listener. The new version will go to provide an input port to send at this listener multiple subscriptions. **PLEASE NOTE THAT ALL THAT YOU CAN DO IN MQTT CAN BE DONE IN ORION BROKER NGSI. Moreover, Orion broker is authenticated, in SSO, provides JSON, etc. This node-red block allows you to subscribe to a topic / device and get event driven actions on IoT App directly.** Please manage the error in output.	Snap4city
	**READ from STORAGE** Query call to Smart City API to get any information about a SURI, ServiceURI. There are many other Nodes which can be used to pose Smart City API queries in very simple manner and recover vectors of ServiceURIs. Please manage the error in output.	Snap4city

Get Data from Storage		**SEARCH on STORAGE** To perform queries on the storage to obtain a list of ServiceURI. The nodes of this family can allow you to perform searching queries by filtering for distance, area, subnature/category, values of attributes, time period, etc. Please manage the error in output.	Snap4city
Send email		Send email. With other nodes you can send Telegram, SMS, etc.	standard
Gen/access to HTTP, HTML pages		To send a **REST CALL** (get, post, etc.). Please USE THIS NODE ONLY for the access at external API and not to access at the Snap4City API for which a lot of MicroServices are accessible as NODEs/Blocks in the Processing Logic and they are simpler to be used and ready to use. Please manage the error in output.	standard
Monitor messages		A block which is printing on **debug** view the data JSON passed in its input. Please note that the node can be tuned to provide only msg.payload or the full JSON message, change configuration of the node.	standard
Stream Delay, limiting rate		A node to insert a delay to each message arriving, or to limit the rate of messages in output. In some cases, the node creates a buffer of messages regularizing the rate in output if the rate in input is greater in some moments.	standard

examples

USage To be trained

	A block which is printing on **debug** view the data JSON passed in its input. Please note that the node can be tuned to provide only msg.payload or the full JSON message, change configuration of the node.	standard
	To **create an Entity Instance** (device instance) from a model prepared on Entity Directory (IoT Directory).	Snap4city
	To **change the ownership** of an Entity Instance (IoT Device).	Snap4city
	To **delegate a certain Entity Instance** (IoT Device) to some other user for which you have to know the Nickname. Delegations can be: Read access, Read write, Modify (to modify the Entity Instance structure)	Snap4city
	To **show something on Snap4City dashboard** with a single content widget (one of the simplest widgets). A large set of dashboard nodes/widgets to send and retrieve data to/from dashboards are provided. This specific Nodes allows to send on dashboard HTML formatted messages with some limitations. Full HTTP widget is also accessible. See in the following section for the Full list of Nodes for	Snap4city
	**MQTT broker listener**, to receive messages from the Broker. Another similar node can be used to send MQTT messages to the MQTT broker. This node allows to perform a subscription to a topic of the MQTT broker.	standard
	**DATA ANALYTICS** Request performed on a Container including a Python data analytics, which is loaded into the node and the container is created at the first Deploy of the Processing Logic. Similar Approach is performed for RStudio Data Analytics.	Snap4city
	**SPLIT:** This block takes in input a buffer, or an array, or an object and split it on a set of messages in output, for each line in the buffer, each element of the array, each element in the object, respectively.	standard
	**JOIN:** This block takes in input a set of messages and join/merge them into a single message (string, buffer, array or object, etc.), on the basis of specific criteria.	standard

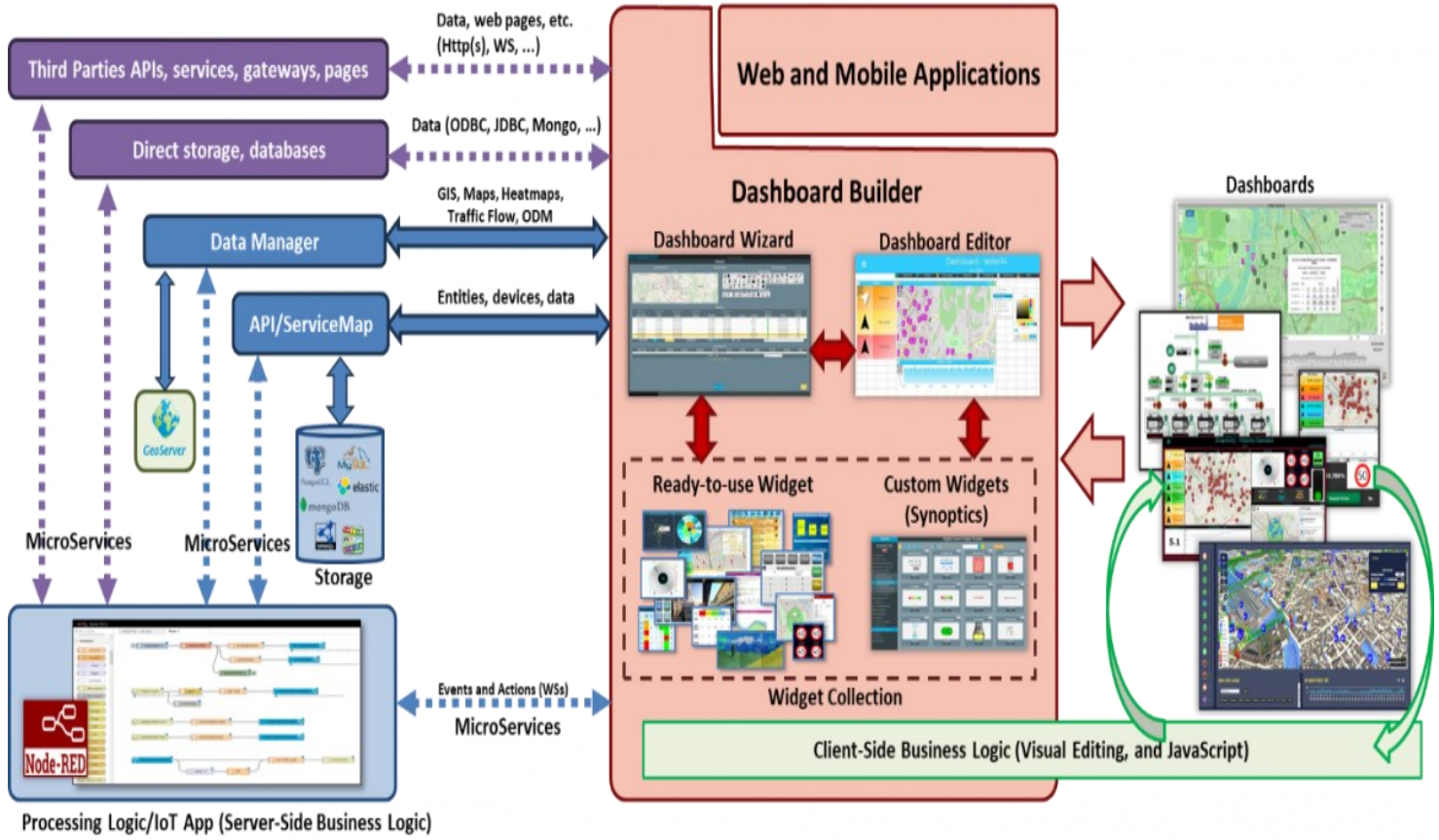
Esempi Data Ingestion

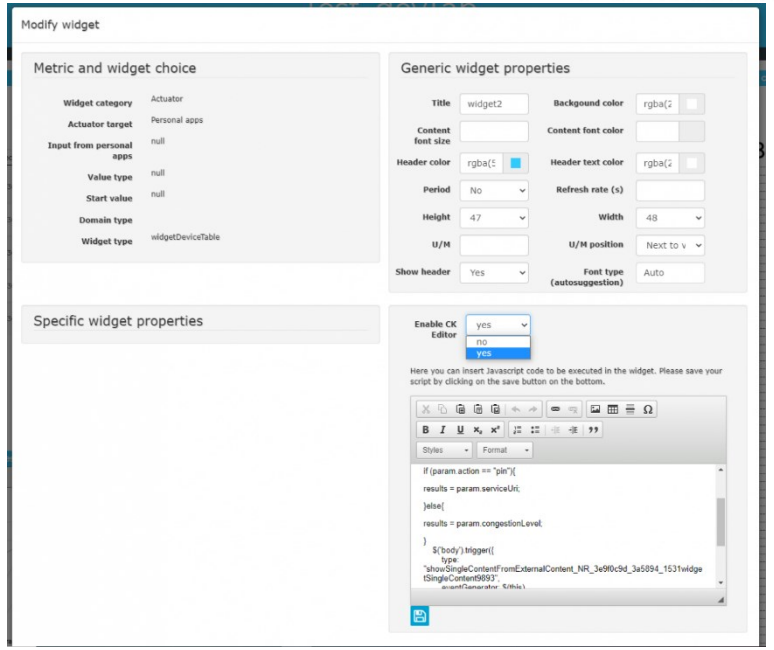
CSBLTutorial

CSBL

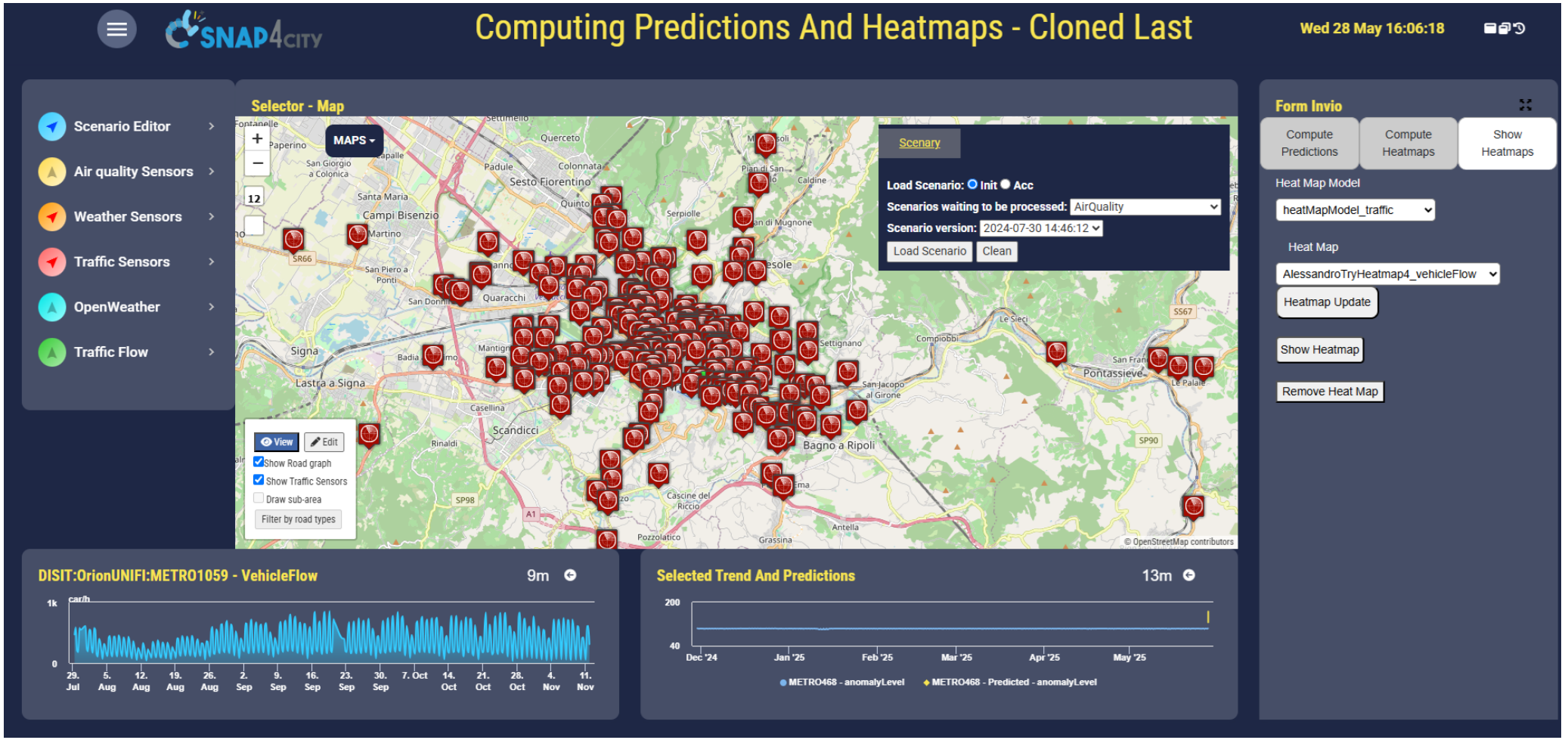
Client-Side Business Logic on Snap4city.org platform

Manual: https://www.snap4city.org/download/video/Snap4Tech-Development-Life-Cycle.pdf





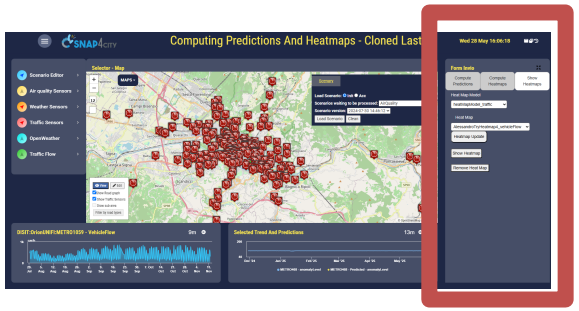
Computing Predictions and Heatmaps



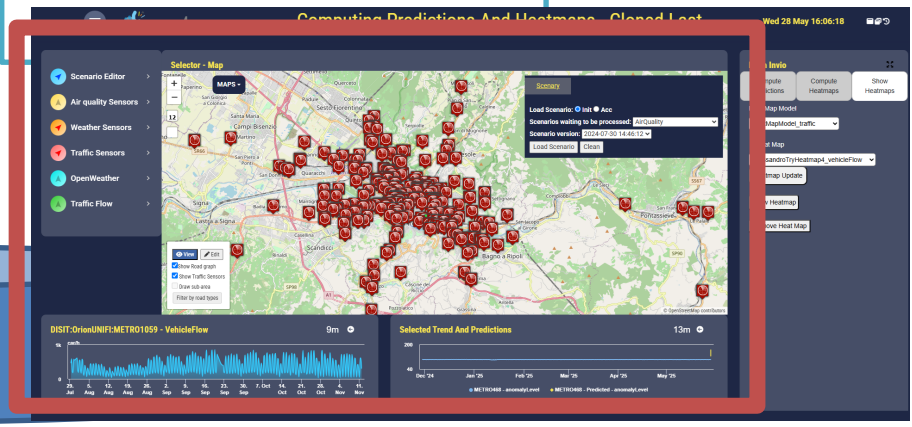
CSBL

API PYTHON

CSBL



Storage



Javascript Request

```javascript
$.ajax({
  url: 'https://api.example.com/data',
  type: 'POST',
  contentType: 'application/json',
  data: JSON.stringify({
    name: 'John',
    age: 30
  }),
  headers: {
    'Authorization': 'Bearer YOUR_ACCESS_TOKEN'
  },
  success: function(response) {
    console.log('Success:', response);
  },
  error: function(xhr, status, error) {
    console.error('Error:', error);
  }
});
```

url: Target API endpoint.

type: 'POST': Method of the request.

contentType: Tells the server the format of the data (application/json).

data: Payload sent to the server (must be stringified).

headers: Include Authorization if needed.

success: Callback for a successful response.

error: Callback for failures.

API Flask

Introduzione a Flask

Cos'è Flask?

- Flask è un micro web framework per Python.
- Leggero e flessibile, ideale per creare API e applicazioni web.
- Basato su Werkzeug (server) e Jinja2 (templating engine).

Caratteristiche principali:

- Minimalista: solo il necessario per iniziare.
- Estendibile con plugin e librerie.
- Ottimo per progetti piccoli o medi.

Componenti principali:

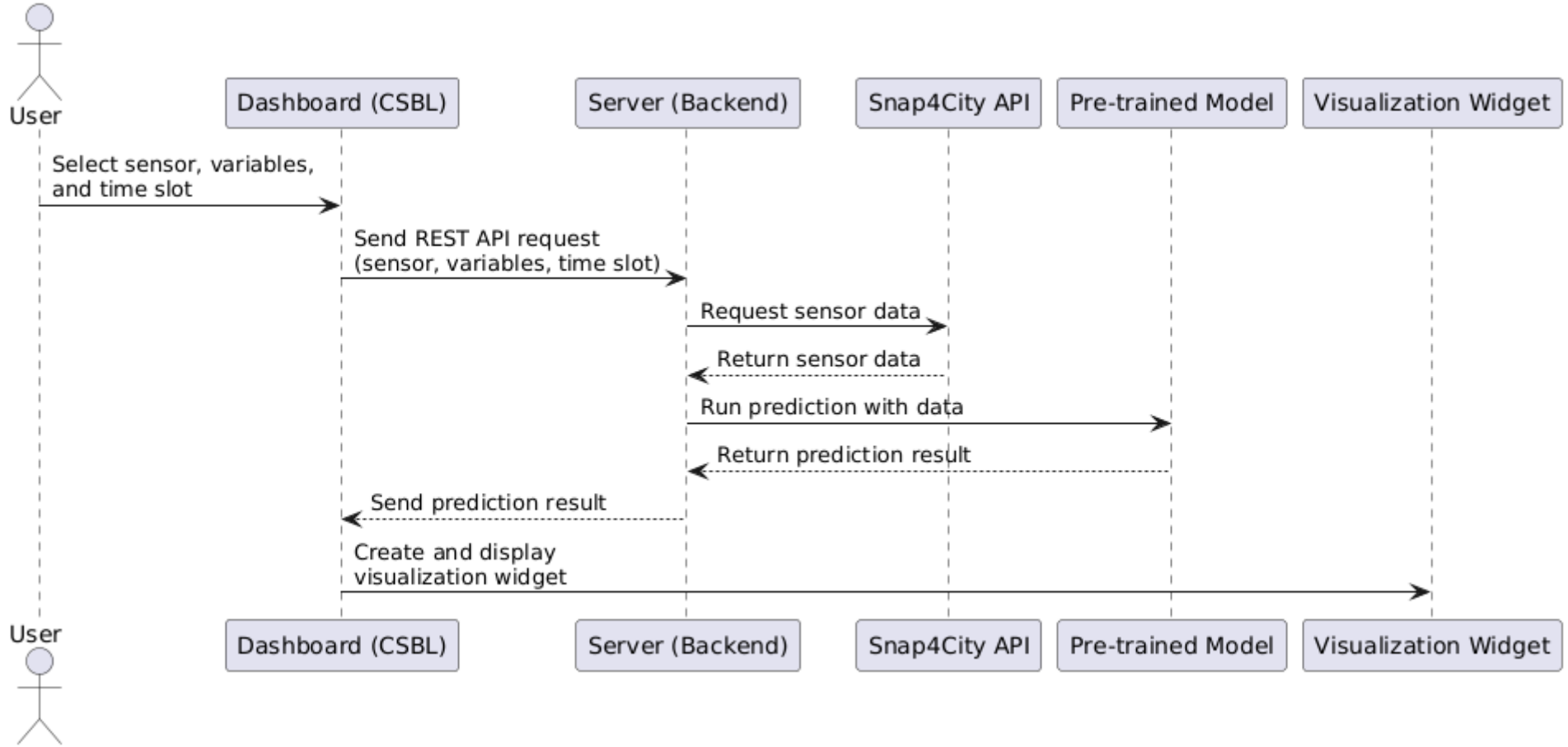
- Routing: associa URL a funzioni
- Request/Response: gestisce dati da form, JSON, URL
- Struttura modulare per grandi progetti
- Tante estensioni utili.

```python
from flask import Flask

app = Flask(__name__)

@app.route("/")
def home():
    return "Ciao, Flask multi-thread!"

if __name__ == "__main__":
    app.run(debug=True, threaded=True)
```

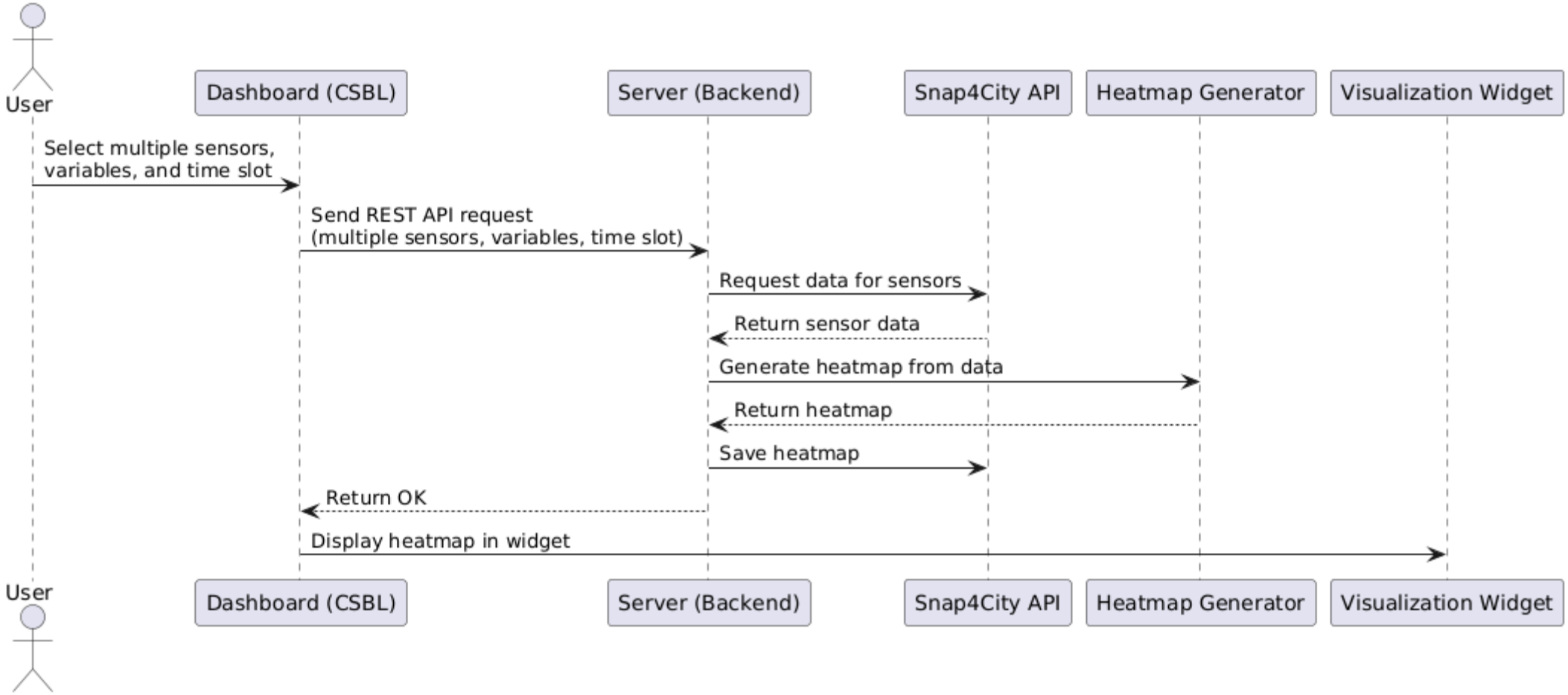
threaded=True
Abilita la gestione multi-thread:
ogni richiesta viene gestita in un thread separato.

Alternativa: processes=n per gestione con multiprocessi, ma meno comune per Flask in sviluppo.

Prediction


User
Dashboard (CSBL)
Server (Backend)
Snap4City API
Pre-trained Model
Visualization Widget
Select sensor, variables, and time slot
Send REST API request (sensor, variables, time slot)
Request sensor data
Return sensor data
Run prediction with data
Return prediction result
Send prediction result
Create and display visualization widget
User
Dashboard (CSBL)
Server (Backend)
Snap4City API
Pre-trained Model
Visualization Widget

Heatmap


User
Dashboard (CSBL)
Server (Backend)
Snap4City API
Heatmap Generator
Visualization Widget
Select multiple sensors, variables, and time slot
Send REST API request (multiple sensors, variables, time slot)
Request data for sensors
Return sensor data
Generate heatmap from data
Return heatmap
Save heatmap
Return OK
Display heatmap in widget
User
Dashboard (CSBL)
Server (Backend)
Snap4City API
Heatmap Generator
Visualization Widget

Computing Predictions and Heatmaps

https://www.snap4city.org/dashboardSmartCity/view/Gea-Night.php?iddasboard=NDYwNA==