IoT NodeRED Maps development

Submitted by: Dipen Gaihre

This document details the successful implementation of an advanced feature in the Python-NodeRED-WorldMap application, demonstrating the system's ability to track and visualize multiple distinct assets simultaneously.

Goal

To upgrade the basic exercise by adding a second, independently moving asset and styling each asset differently (color, icon) using Node-RED logic.

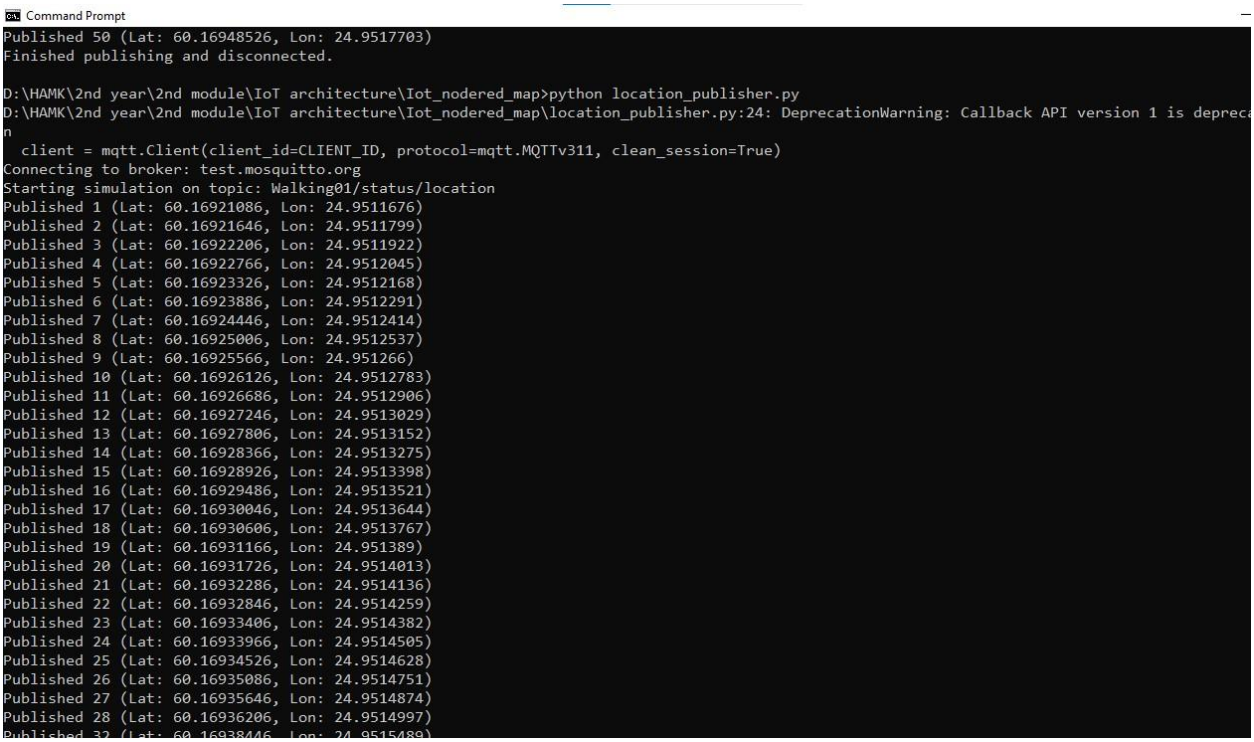
Implementation Steps and Evidence

The solution required modifying the data publishers (Python) and implementing conditional routing in Node-RED using the switch node.

1. Added a Second "Driver" (Python)

We copied our original Python code and changed a few lines to create a new, separate tracker.

- **Original Tracker:** Named "DipenLocationTracker" (our person).
- **New Tracker:** Named **"RescueTeamVehicle"** (our car), starting in a different spot.


Command Prompt
Published 50 (Lat: 60.16948526, Lon: 24.9517703)
Finished publishing and disconnected.
D:\HAMK\2nd year\2nd module\IoT architecture\Iot_nodered_map>python location_publisher.py
D:\HAMK\2nd year\2nd module\IoT architecture\Iot_nodered_map\location_publisher.py:24: DeprecationWarning: Callback API version 1 is deprec
n
client = mqtt.Client(client_id=CLIENT_ID, protocol=mqtt.MQTTv311, clean_session=True)
Connecting to broker: test.mosquitto.org
Starting simulation on topic: Walking01/status/location
Published 1 (Lat: 60.16921086, Lon: 24.9511676)
Published 2 (Lat: 60.16921646, Lon: 24.9511799)
Published 3 (Lat: 60.16922206, Lon: 24.9511922)
Published 4 (Lat: 60.16922766, Lon: 24.9512045)
Published 5 (Lat: 60.16923326, Lon: 24.9512168)
Published 6 (Lat: 60.16923886, Lon: 24.9512291)
Published 7 (Lat: 60.16924446, Lon: 24.9512414)
Published 8 (Lat: 60.16925006, Lon: 24.9512537)
Published 9 (Lat: 60.16925566, Lon: 24.951266)
Published 10 (Lat: 60.16926126, Lon: 24.9512783)
Published 11 (Lat: 60.16926686, Lon: 24.9512906)
Published 12 (Lat: 60.16927246, Lon: 24.9513029)
Published 13 (Lat: 60.16927806, Lon: 24.9513152)
Published 14 (Lat: 60.16928366, Lon: 24.9513275)
Published 15 (Lat: 60.16928926, Lon: 24.9513398)
Published 16 (Lat: 60.16929486, Lon: 24.9513521)
Published 17 (Lat: 60.16930046, Lon: 24.9513644)
Published 18 (Lat: 60.16930606, Lon: 24.9513767)
Published 19 (Lat: 60.16931166, Lon: 24.951389)
Published 20 (Lat: 60.16931726, Lon: 24.9514013)
Published 21 (Lat: 60.16932286, Lon: 24.9514136)
Published 22 (Lat: 60.16932846, Lon: 24.9514259)
Published 23 (Lat: 60.16933406, Lon: 24.9514382)
Published 24 (Lat: 60.16933966, Lon: 24.9514505)
Published 25 (Lat: 60.16934526, Lon: 24.9514628)
Published 26 (Lat: 60.16935086, Lon: 24.9514751)
Published 27 (Lat: 60.16935646, Lon: 24.9514874)
Published 28 (Lat: 60.16936206, Lon: 24.9514997)

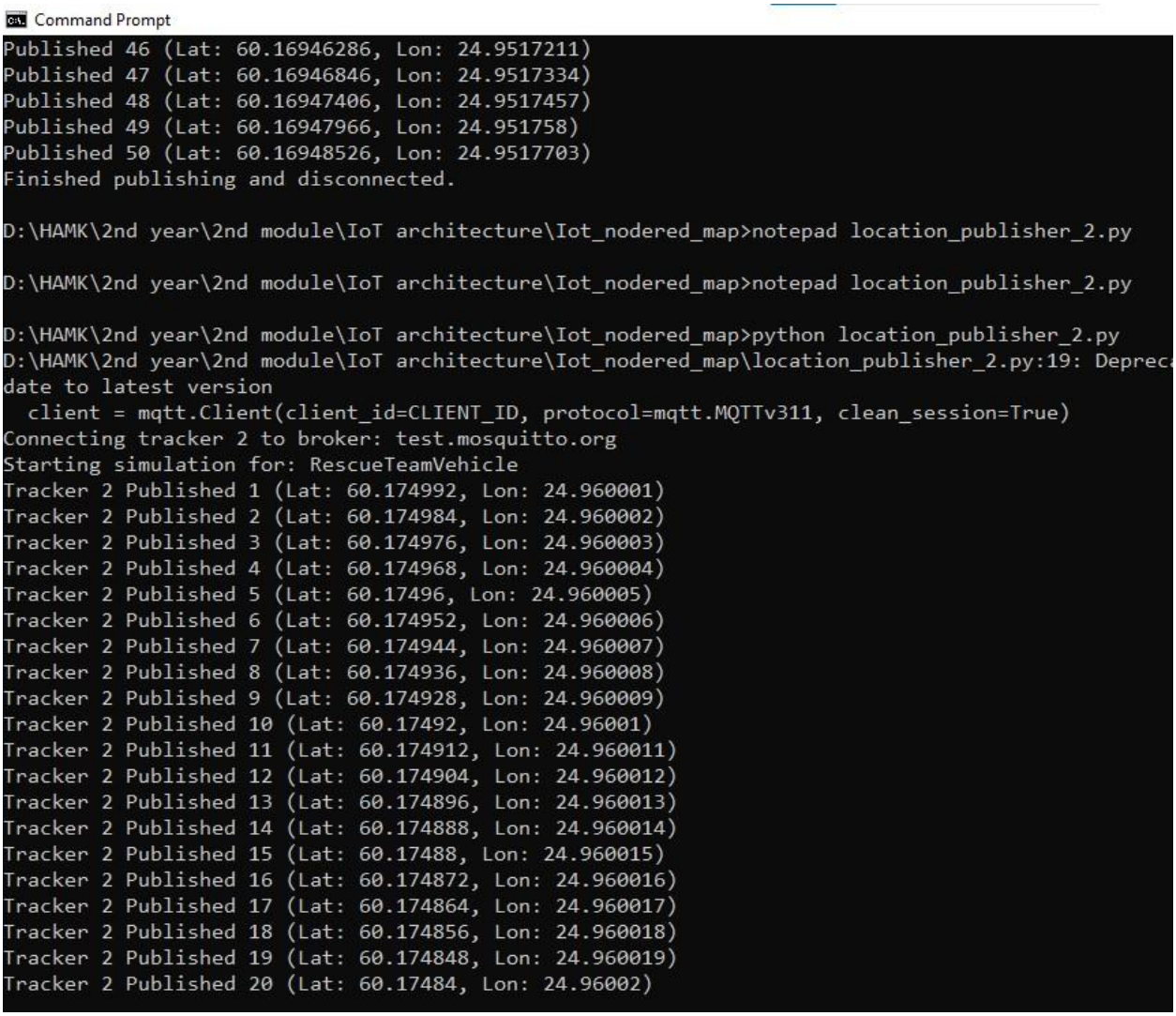


Fig: Python Scripts Running

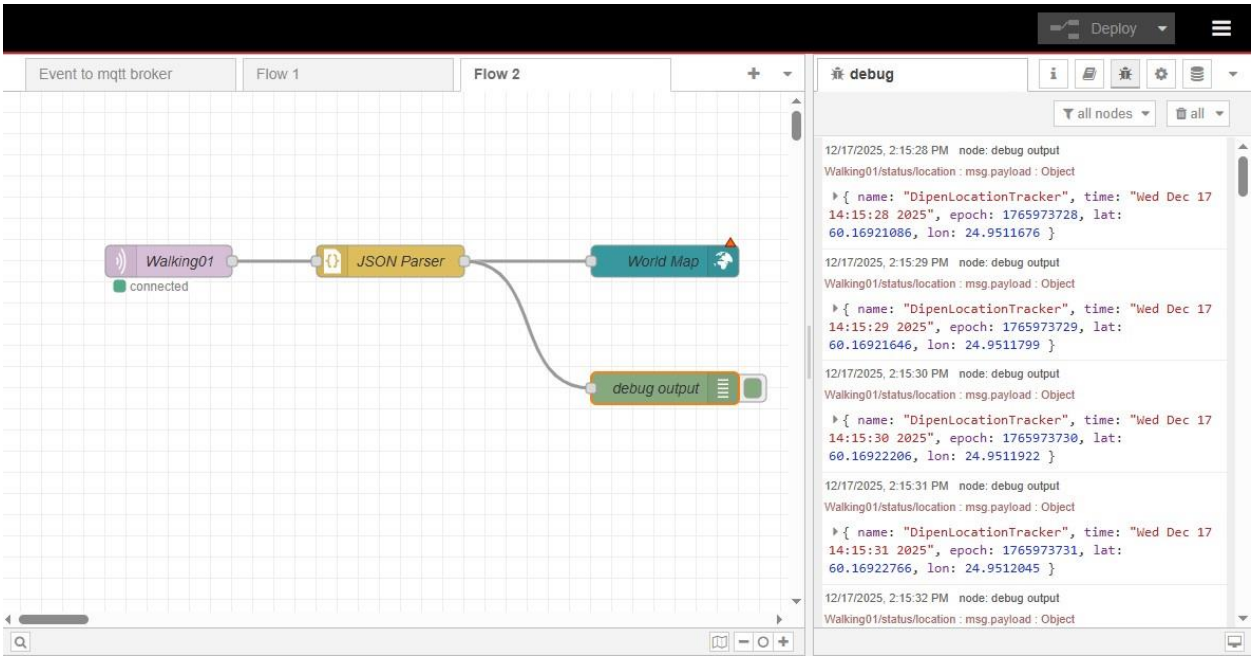


Fig: Debug Output

2. Created a "Traffic Cop" (Node-RED Switch)

Because our simple map couldn't handle the "if/then" logic (if it's the car, make it red), we installed a **switch node** to act as a traffic cop, directing each message to the right styling station.

- **The Cop's Rule:** If the name is "RescueTeamVehicle," send it left. If the name is "DipenLocationTracker," send it right.

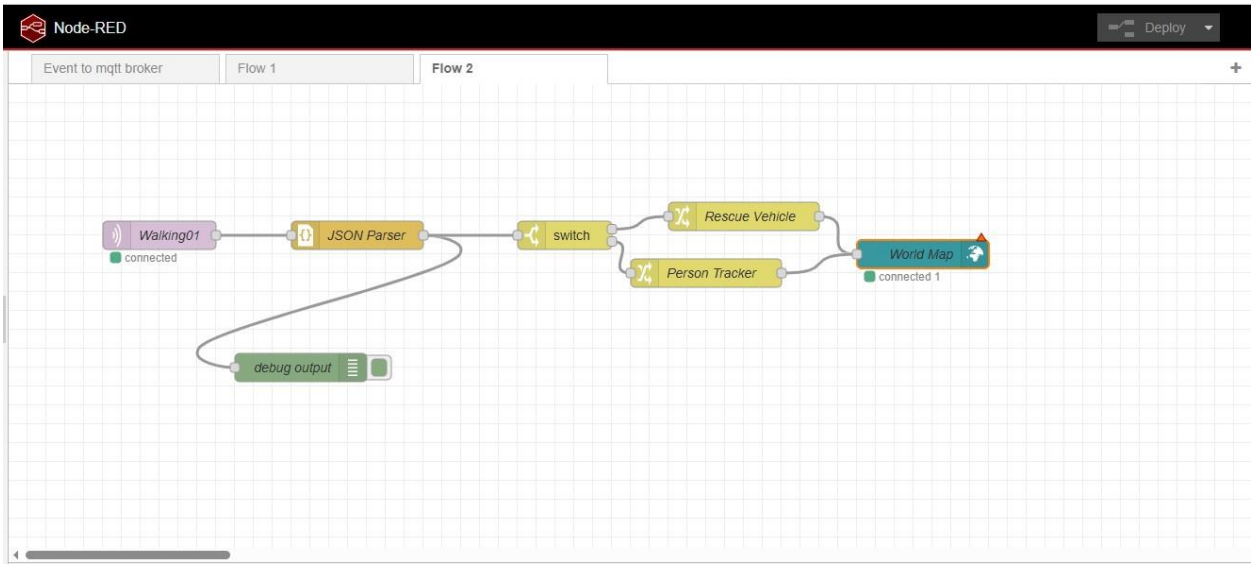
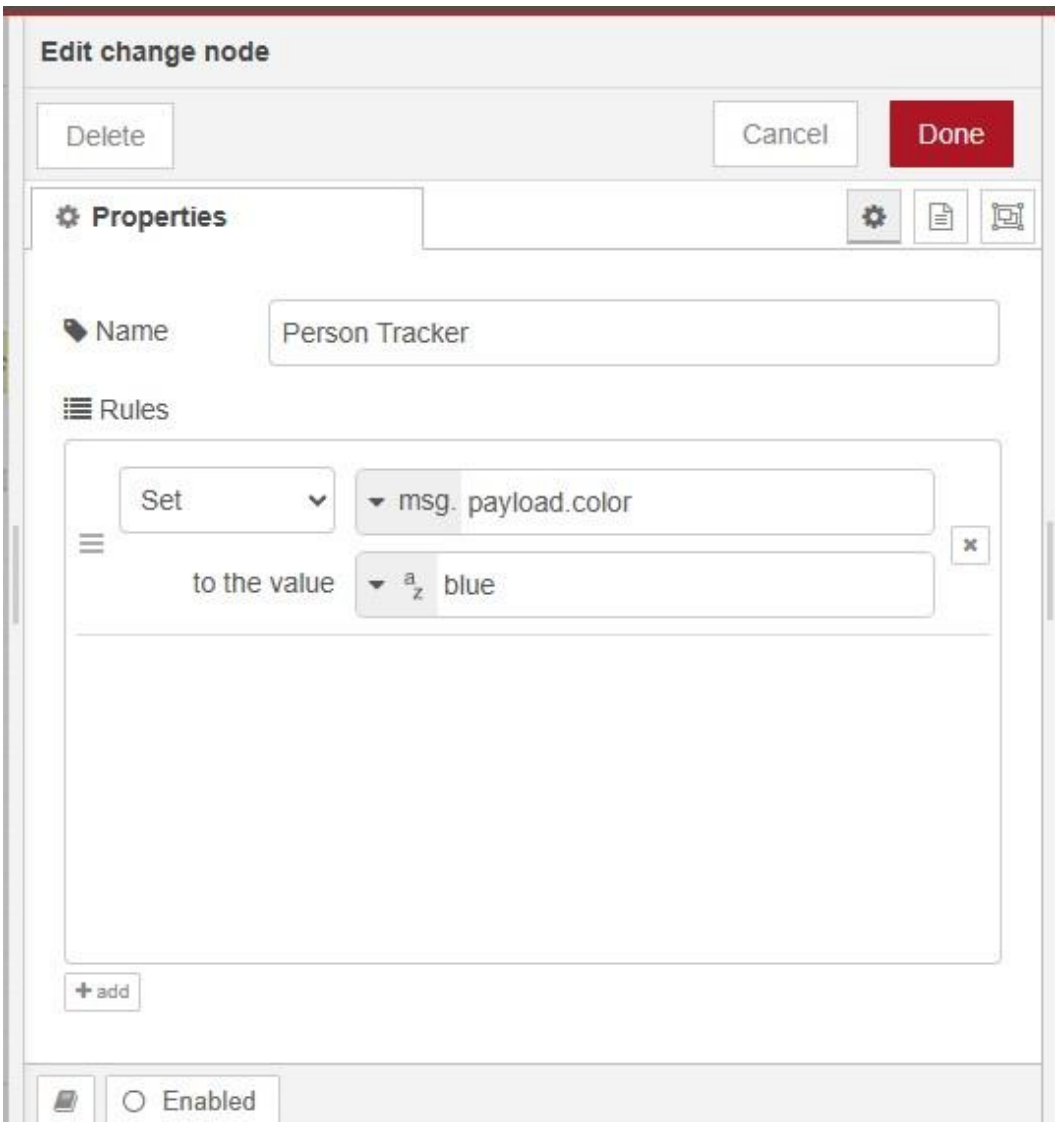


Fig: Node-RED Flow Structure

3. Assigned Team Uniforms (Node-RED Change)

We added two small **change nodes** at the end of the lines—these are the "uniform stations" that add color and icons.

- **Car's Uniform (Path 1):** We changed the marker color to **Red** and gave it a **Car icon**.
- **Person's Uniform (Path 2):** We changed the marker color to **Blue**.


Edit change node
Delete
Cancel
Done
Properties
Name
Person Tracker
Rules
Set
msg.
payload.color
to the value
blue
add
Enabled

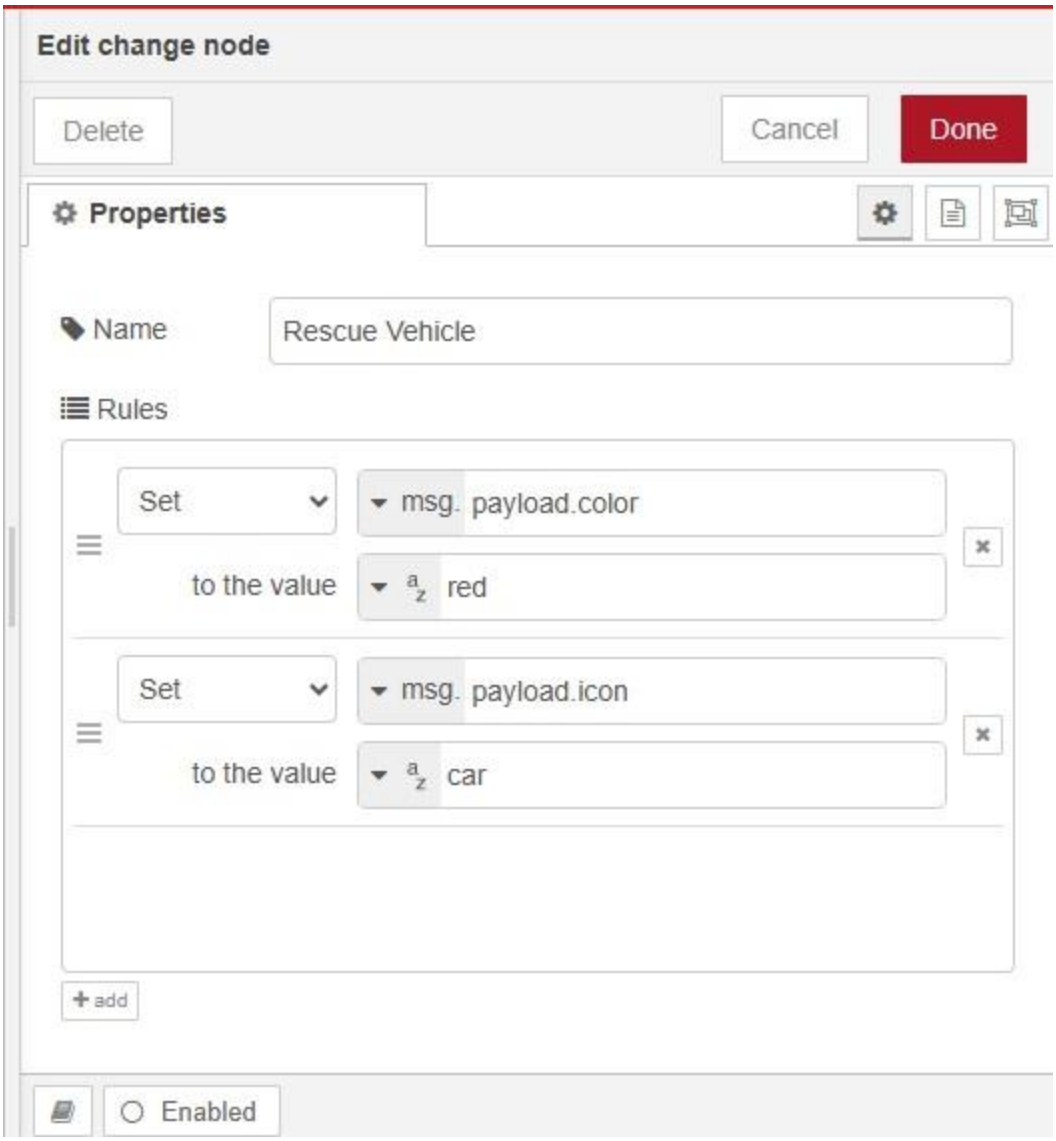


Fig: Rescue Vehicle Node Config and Person Tracker Node Config

4. The Final Result

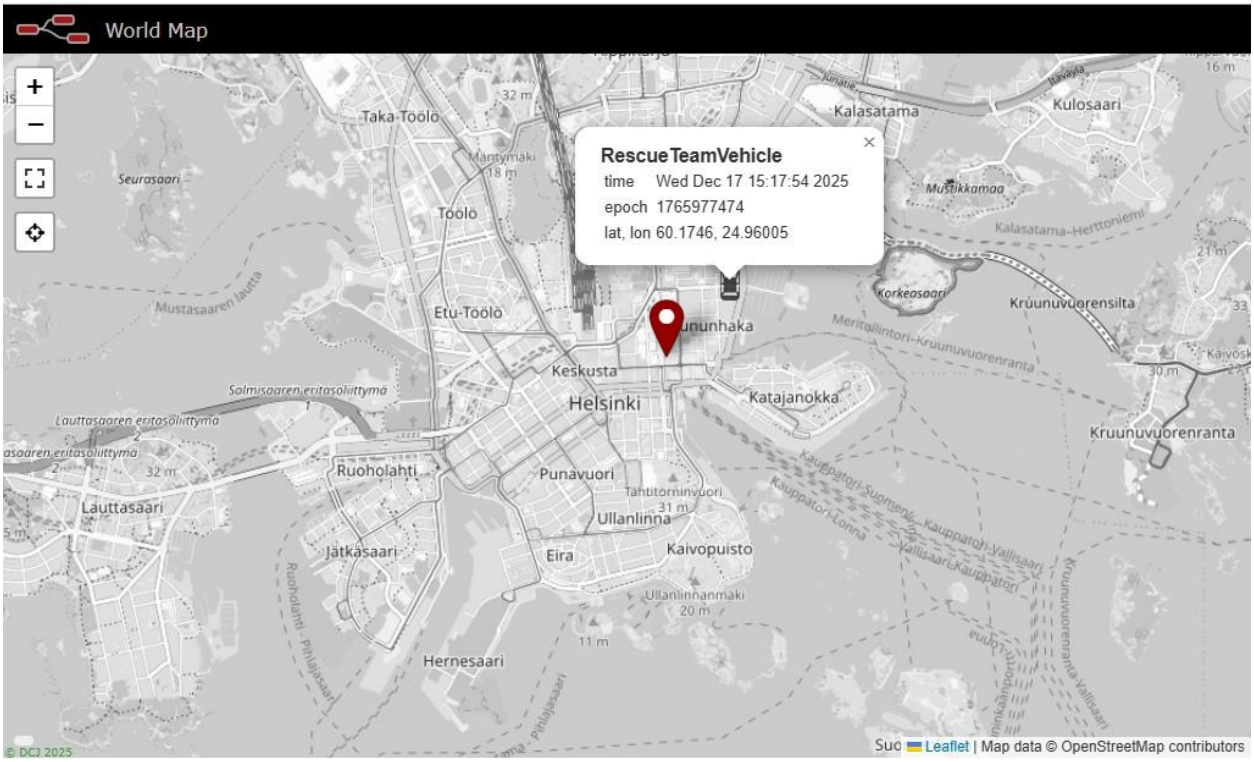


fig: Final World Map

The final map showing **two separate, moving markers**: a **red car** and a **blue person** marker.