



Πανεπιστήμιο Δυτικής Αττικής

Σχολή Μηχανικών

Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

ΠΜΣ «Προηγμένες Τεχνολογίες Υπολογιστικών Συστημάτων»

Συστήματα ευφυούς διαχείρισης πόρων και υποδομών στο Διαδίκτυο των Αντικειμένων

Διδάσκων: Παναγιώτης Καρκαζής

Υλοποίηση Εφαρμογής Smart Home:

Έξυπνος θερμοστάτης και ανιχνευτής καπνού

mscacs24009 Βασίλειος Μητσοκάπας

Αιγάλεω, 06 Ιουνίου, 2025

Κατάλογος Περιεχομένων

1. Αρχεία Εργασίας.....	3
2. Περιγραφή της Εργασίας.....	4
3. Χρησιμοποιούμενο Hardware.....	5
4. Εγκατάσταση Περιβάλλοντος	8
5. Περιγραφή Αρχείων.....	12
6. Node-RED Flows	15
7. Node-RED Dashboard	24



1. Αρχεία Εργασίας

- Βίντεο παρουσίασης: <https://www.youtube.com/watch?v=a4UhWK637Yc>
- Project GitHub: https://github.com/mbasil84/IoT_fire_project

2. Περιγραφή της Εργασίας

Η παρούσα IoT εφαρμογή έχει σχεδιαστεί για να λειτουργεί ως έξυπνος ανιχνευτής πυρκαγιάς και θερμοστάτης, υλοποιώντας τα εξής:

➤ Χρήση αισθητήρων:

- **DHT22:** μετράει θερμοκρασία και υγρασία του περιβάλλοντος.
- **MQ-5:** ανιχνεύει την παρουσία καπνού στον αέρα μέσω της ψηφιακής του εξόδου.

➤ Δημοσίευση δεδομένων μέσω MQTT:

Η συσκευή IoT (Raspberry Pi) στέλνει περιοδικά (ανά 5 δευτερόλεπτα) ένα JSON μήνυμα στο topic **fire_status** το οποίο περιλαμβάνει:

- **device_id:** το όνομα της συσκευής
- **temperature:** τιμή σε βαθμούς °C
- **humidity:** τιμή σε ποσοστό %
- **smoke:** Boolean (true/false)

➤ Οπτικοποίηση και έλεγχος μέσω Node-RED:

Ο server Node-RED λειτουργεί ως επεξεργαστής των μετρήσεων και ταυτόχρονα δίνει εντολές μέσω MQTT:

- Ο χρήστης βλέπει θερμοκρασία/υγρασία σε **gauges** και **charts**.
- Μπορεί να ορίσει επιθυμητή θερμοκρασία μέσω ενός **slider**.
- Το σύστημα αποφασίζει αν θα ενεργοποιήσει τη θέρμανση ή όχι (σύγκριση τρέχουσας θερμοκρασίας με την επιθυμητή και ενεργοποίηση ή απενεργοποίηση ρελέ).
- Ενεργοποιούνται **LEDs** ανάλογα με τη θερμοκρασία.
- Ενεργοποιείται **buzzer** σε περίπτωση καπνού.
- Ο χρήστης βλέπει ποιο LED είναι ενεργό, αν ανιχνεύεται καπνός στο χώρο και αν είναι ενεργοποιημένη η θέρμανση.

Η εφαρμογή είναι ιδανική για monitoring ενός χώρου, καθώς παρέχει αυτόματη ειδοποίηση σε επικίνδυνες συνθήκες (καπνός) ενώ αλληλεπιδρά και με το σύστημα θέρμανσης.

3. Χρησιμοποιούμενο Hardware

Η εφαρμογή υλοποιήθηκε με ένα Raspberry Pi 5, με τη χρήση breadboard και GPIO extension.

Αναλυτικά:

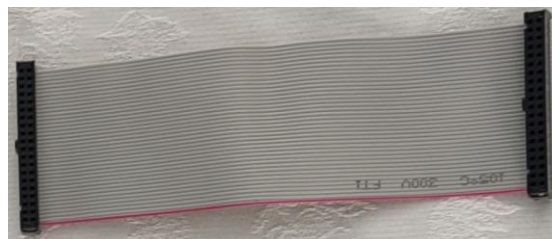
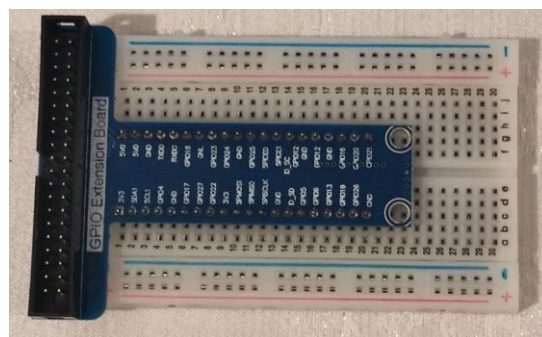
➤ Raspberry Pi 5

- Τρέχει τους Python publishers/subscribers.
- Έχει σύνδεση σε τοπικό δίκτυο για επικοινωνία με τον MQTT broker και τον Node-RED.



➤ GPIO Extension Board + Breadboard

- Διευκολύνει τη σύνδεση των ακίδων GPIO σε breadboard.
- Προσφέρει καθαρή και σταθερή καλωδίωση των εξαρτημάτων.



➤ **Αισθητήρας θερμοκρασίας/υγρασίας: DHT22**

- Παρέχει ακριβείς μετρήσεις θερμοκρασίας ($^{\circ}\text{C}$) – υγρασίας (%).
- Ανθεκτικός για μετρήσεις εσωτερικού χώρου.



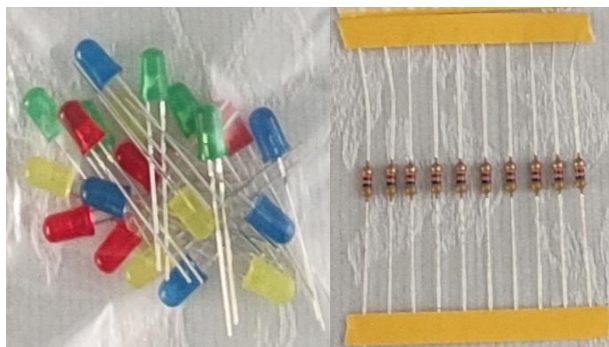
➤ **Αισθητήρας καπνού: MQ-5**

- Χρησιμοποιείται η ψηφιακή έξοδος (D0).
- Παράγει True όταν ανιχνευτεί καπνός.



➤ **4 LEDs (μπλε, πράσινο, κίτρινο, κόκκινο) και αντιστάσεις**

- Παρέχουν οπτική αναπαράσταση της θερμοκρασίας:
 - $<20^{\circ}\text{C} \rightarrow$ Μπλε
 - $20-25^{\circ}\text{C} \rightarrow$ Πράσινο
 - $25-30^{\circ}\text{C} \rightarrow$ Κίτρινο
 - 30°C (ή σε ανίχνευση καπνού) \rightarrow Κόκκινο
- Αντιστάσεις σε σειρά με LEDs για προστασία (220Ω).



➤ **Buzzer (active)**

- Ενεργοποιείται αυτόματα όταν υπάρχει καπνός.



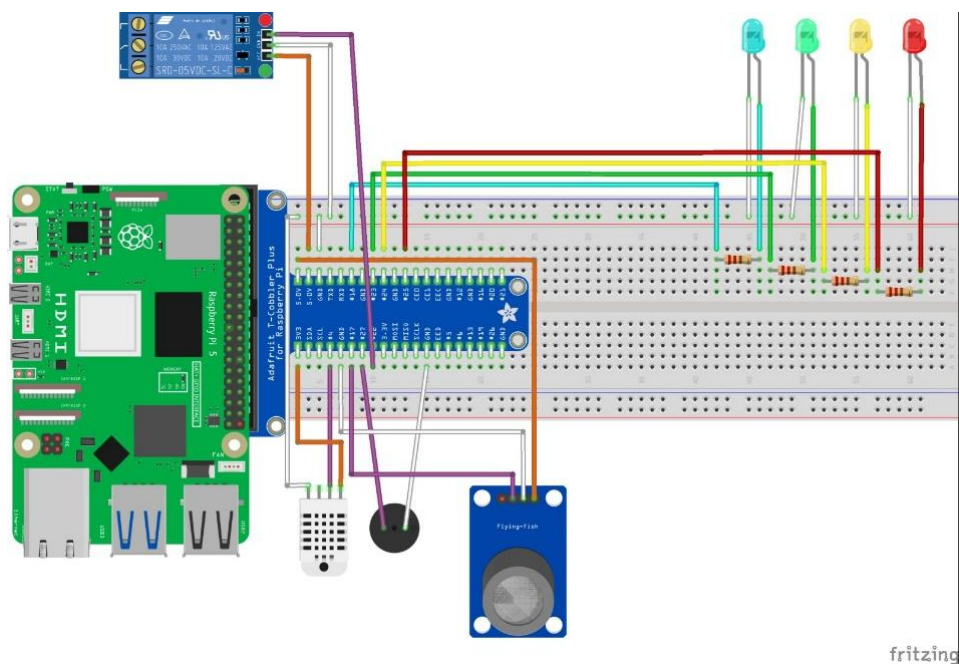
➤ **Ρελέ 1 καναλιού**

- Αντιπροσωπεύει το σύστημα θέρμανσης.
- Ενεργοποιείται από τον Node-RED όταν η πραγματική θερμοκρασία είναι μικρότερη από την επιθυμητή.



➤ **Διάγραμμα (Fritzing)**

Το παρακάτω διάγραμμα αποτυπώνει καθαρά όλες τις συνδέσεις:



4. Εγκατάσταση Περιβάλλοντος

Η υλοποίηση βασίζεται σε δύο συνδυαστικά περιβάλλοντα:

1. Python Virtual Environment στο Raspberry Pi (scripts)
2. Docker Containers (MQTT broker και Node-RED)

και επιτρέπει τον διαχωρισμό συστήματος:

- Raspberry → αισθητήρες και GPIO
- Laptop → broker και Node-RED

➤ Python Environment (Raspberry Pi)

- Δημιουργία εικονικού περιβάλλοντος Python
 - ❖ `python3 -m venv venv`
- Ενεργοποίηση περιβάλλοντος
 - ❖ `source venv/bin/activate`
- Εγκατάσταση εξαρτήσεων
 - ❖ `pip install -r requirements.txt`

requirements.txt:

```
paho-mqtt  
gpiozero  
adafruit-circuitpython-dht  
adafruit-blinka
```

➤ Docker & WSL (Windows)

- Εγκατάσταση Docker Desktop
- Ενεργοποίηση WSL 2 ως backend
- Δημιουργία φακέλου με docker-compose.yml

docker-compose.yml:

```
version: '3'
```


services:

mosquitto:

image: eclipse-mosquitto

container_name: mosquitto

ports:

- "1883:1883"

volumes:

- ./mosquitto/config:/mosquitto/config

- ./mosquitto/data:/mosquitto/data

nodered:

image: nodered/node-red

container_name: nodered

ports:

- "1880:1880"

restart: unless-stopped

mosquitto.conf:

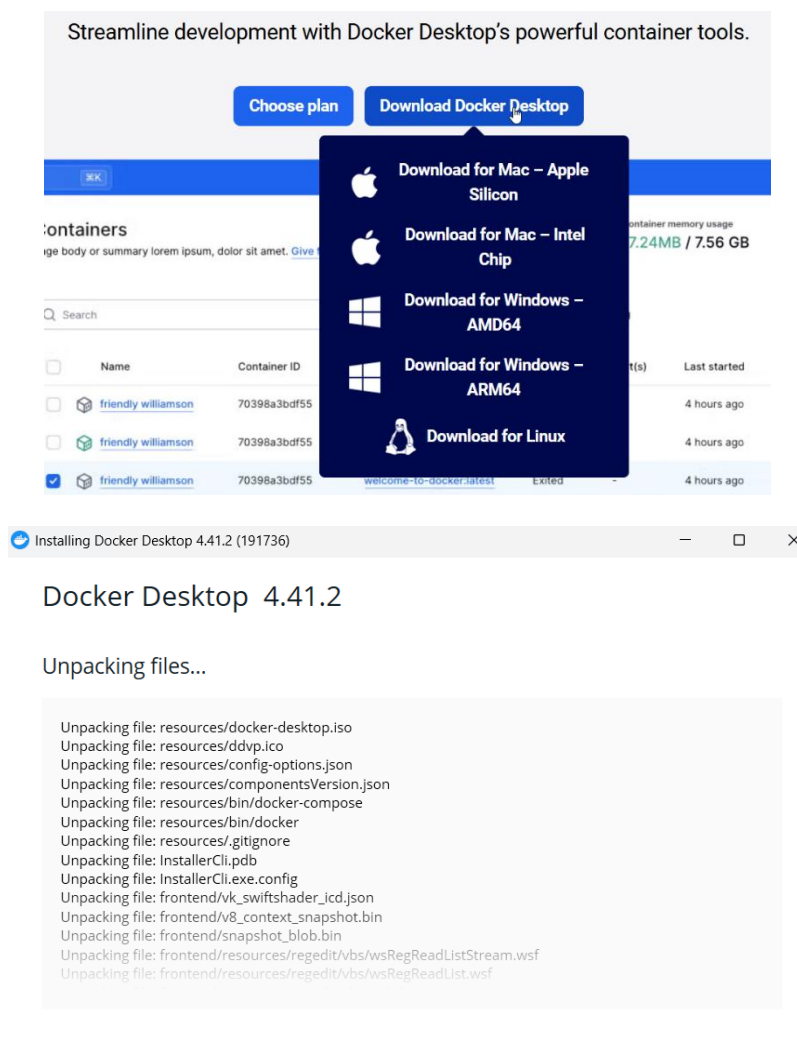
persistence true

persistence_location /mosquitto/data/

log_dest stdout

allow_anonymous true

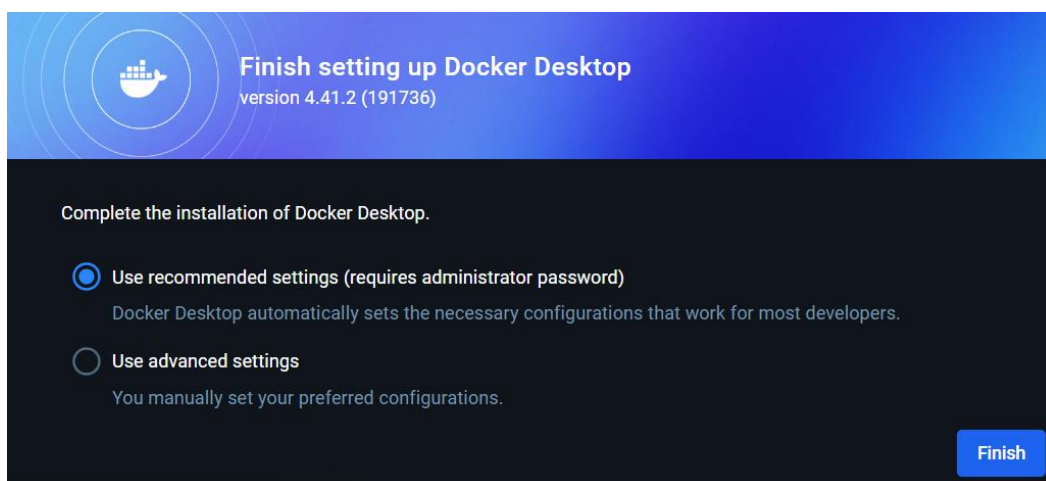
listener 1883



Docker Desktop 4.41.2

Unpacking files...

```
Unpacking file: resources/docker-desktop.iso
Unpacking file: resources/ddvp.ico
Unpacking file: resources/config-options.json
Unpacking file: resources/componentsVersion.json
Unpacking file: resources/bin/docker-compose
Unpacking file: resources/bin/docker
Unpacking file: resources/gitignore
Unpacking file: InstallerCli.pdb
Unpacking file: InstallerCli.exe.config
Unpacking file: frontend/vk_swiftshader_icd.json
Unpacking file: frontend/v8_context_snapshot.bin
Unpacking file: frontend/snapshot_blob.bin
Unpacking file: frontend/resources/regedit/vbs/wsRegReadListStream.wsf
Unpacking file: frontend/resources/regedit/vbs/wsRegReadList.wsf
```



```
C:\WINDOWS\system32\wsl.exe
Windows Subsystem for Linux must be updated to the latest version to proceed. You can update by running 'wsl.exe --update'.
For more information please visit https://aka.ms/wslinstall

Press any key to install Windows Subsystem for Linux.
Press CTRL-C or close this window to cancel.
This prompt will time out in 60 seconds.
Downloading: Windows Subsystem for Linux 2.4.13
Installing: Windows Subsystem for Linux 2.4.13
Windows Subsystem for Linux 2.4.13 has been installed.
The operation completed successfully.
Checking for updates.
The most recent version of Windows Subsystem for Linux is already installed.
Press any key to exit...
```

- **Εκκίνηση containers:**

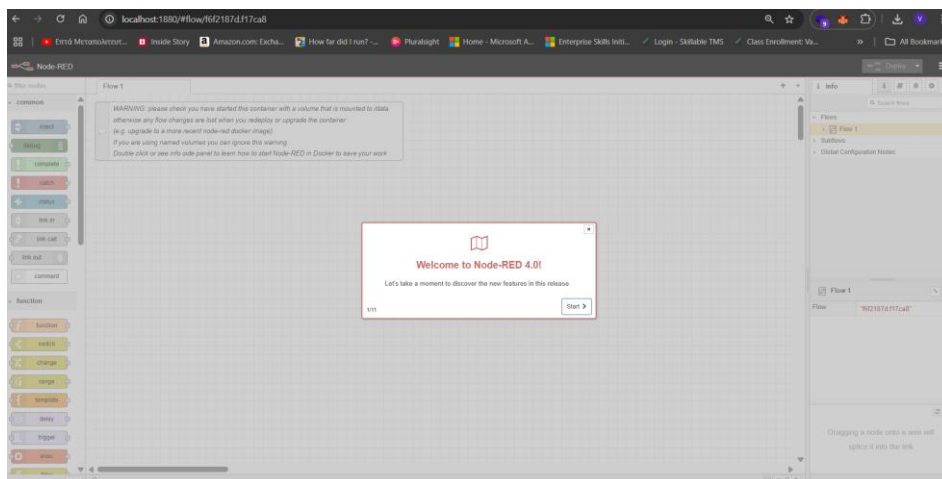
- ❖ *docker compose up -d*

```
Administrator: Windows PowerShell
iot-docker

PS C:\WINDOWS\system32> cd C:\iot-docker
PS C:\iot-docker> docker compose up -d
time="2025-05-10T10:54:38+03:00" level=warning msg="C:\\iot-docker\\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[*] Running 20/21
✔ mosquitto Pulled 5.6s
✔ 86da743658 Pull complete 1.5s
✔ f328c2748685 Pull complete 2.4s
✔ nodored Pulled 30.2s
✔ 4f4b708e154 Pull complete 0.6s
✔ 729f1adceac Pull complete 0.6s
✔ f18232174bc9 extracting 1 s 27.4s
✔ 0a313e080db Pull complete 1.5s
✔ d2c55ab5b31 Pull complete 1.5s
✔ ecc33e081cd Pull complete 1.5s
✔ 8a066e22f58 Pull complete 1.5s
✔ 07809a788e0 Pull complete 15.4s
✔ 3a676a6931f5 Pull complete 1.5s
✔ 42ec265e2954 Pull complete 1.5s
✔ 6c2a9a5911c Pull complete 15.3s
✔ 6c2a9a590e66 Pull complete 1.5s
✔ 0742486c80f Pull complete 1.5s
✔ 0b03a9ac9c5 Pull complete 27.2s
✔ 5dda236ff27 Pull complete 21.4s
✔ a13ef429ac3d Pull complete 1.5s
✔ 6e6c1f00a3 Pull complete 1.5s
[*] Running 5/3
✔ Network iot-docker_default Created 0.1s
✔ Container mosquitto Started 3.2s
✔ Container nodored Started 1.2s
PS C:\iot-docker>
```

- **Επιβεβαίωση:**

- Node-RED UI: <http://localhost:1880>
- MQTT broker: localhost:1883



5. Περιγραφή Αρχείων

Η λειτουργία του συστήματος βασίζεται σε δύο κύρια αρχεία Python τα οποία εκτελούνται στο Raspberry Pi και αλληλεπιδρούν με τον Node-RED μέσω MQTT.

➤ **fire_publisher.py**

Λειτουργία: Συλλέγει δεδομένα αισθητήρων και τα στέλνει περιοδικά στον MQTT broker.

- Βιβλιοθήκες: adafruit_dht, gpiozero, paho.mqtt.client, json, time
- Συσκευές εισόδου:
 - DHT22 στον GPIO4 για θερμοκρασία και υγρασία
 - MQ-5 στον GPIO17 (ψηφιακή έξοδος) για καπνό
- MQTT Ρυθμίσεις:
 - Broker IP: 192.168.1.51 (ip του laptop)
 - Port: 1883
 - Topic: fire_status

➤ **iot_subscriber.py**

Λειτουργία: Λαμβάνει εντολές από Node-RED (μέσω MQTT) και ελέγχει τα εξαρτήματα εξόδου (LEDs, buzzer, relay).

Topics που παρακολουθεί:

- led_control: ενεργοποίηση συγκεκριμένου LED.
- buzzer_control: ενεργοποίηση/απενεργοποίηση buzzer.
- heating_control: ενεργοποίηση/απενεργοποίηση relay θέρμανσης.

GPIO αντιστοιχίσεις:

- LEDs:
 - Blue: GPIO18
 - Green: GPIO23
 - Yellow: GPIO24
 - Red: GPIO25
- Buzzer: GPIO27

- Relay: GPIO22

➤ **Αλληλεπίδραση fire_publisher ↔ Node-RED ↔ iot_subscriber**

A. Από Raspberry Pi προς Node-RED

To script fire_publisher.py στέλνει κάθε 5 δευτερόλεπτα ένα MQTT μήνυμα στο topic: fire_status

Αποστολέας: fire_publisher.py
Παραλήπτης: Node-RED
Payload JSON:

```
{  
  "device_id": "sensor_pi_5",  
  "temperature": 27.6,  
  "humidity": 51.3,  
  "smoke": false,  
  "unit": "C"  
}
```

To Node-RED:

- Εμφανίζει τις τιμές σε:
 - Gauges (θερμοκρασία, υγρασία)
 - Charts (ιστορικό)
 - Text nodes (ενδείξεις σε μορφή κειμένου/emoji)
- Λαμβάνει επιθυμητή θερμοκρασία από slider και τη συγκρίνει με msg.payload.temperature.
- Αν διαπιστώσει:
 - Καπνό: ενεργοποιεί το buzzer.
 - Θερμοκρασία < Επιθυμητής: ενεργοποιεί το relay (θέρμανση).
 - Καθορίζει ποιο LED θα ανάψει ανά θερμική ζώνη.

B. Από Node-RED προς Raspberry Pi

To iot_subscriber.py περιμένει εντολές από 3 διαφορετικά MQTT topics:

Topic: led_control

Payload:

```
{ "led": "yellow" }
```

Ενεργοποιεί μόνο το αντίστοιχο LED.

Topic: buzzer_control

Payload:

```
{ "buzzer": true }
```

Ενεργοποιεί το buzzer (ή false για απενεργοποίηση).

Topic: heating_control

Payload:

```
{ "heating": true }
```

Ενεργοποιεί το relay (ή false για απενεργοποίηση).

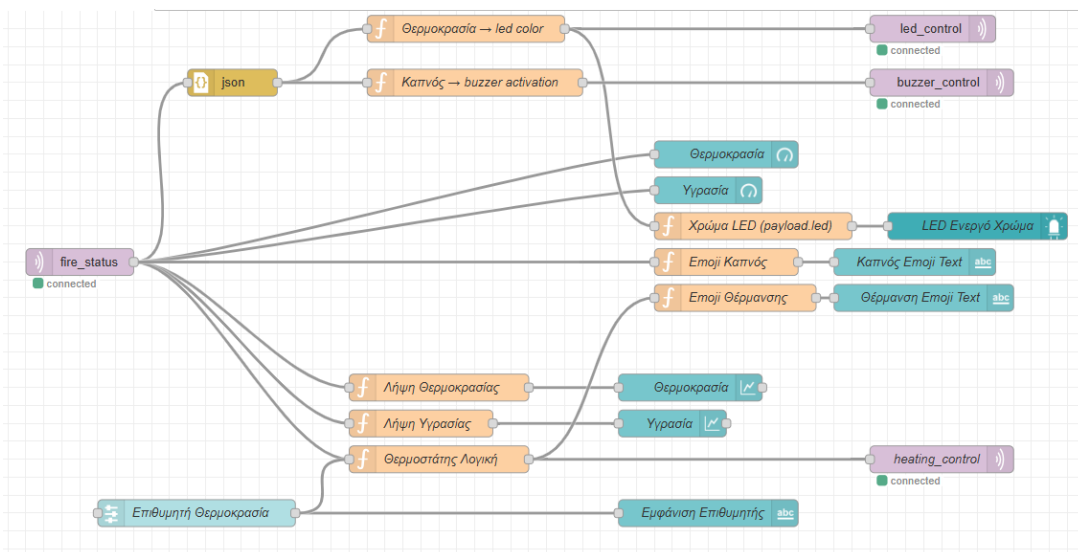
6. Node-RED Flows

Το Node-RED λειτουργεί ως **κεντρικός ελεγκτής λογικής** του συστήματος: λαμβάνει τα δεδομένα των αισθητήρων, τα απεικονίζει, παίρνει αποφάσεις και στέλνει εντολές πίσω στο Raspberry Pi μέσω MQTT.

➤ Επισκόπηση Flow

Η βασική ροή αποτελείται από τα εξής βήματα:

1. **Είσοδος από MQTT (fire_status)**
2. **Ανάλυση JSON**
3. **Διανομή τιμών:**
 - Θερμοκρασία → Gauge, Chart
 - Υγρασία → Gauge, Chart
 - Καπνός → Έλεγχος και ένδειξη ενεργοποίησης buzzer ή μη
4. **LED λογική θερμοκρασίας:**
 - Σύγκριση επιθυμητής θερμοκρασίας με πραγματική
 - Απόφαση ενεργοποίησης ή απενεργοποίησης θέρμανσης
5. **Δημιουργία MQTT εξόδων προς Raspberry Pi**



Edit json node

Delete Cancel Done

⚙ Properties

⊕ Action Always convert to JavaScript Object

⋯ Property msg.payload

📌 Name Name

Edit function node

Delete Cancel Done

⚙ Properties

📌 Name Θερμοκρασία → led color

⚙ Setup On Start On Message On Stop

```

1 let temp = msg.payload.temperature;
2 let smoke = msg.payload.smoke;
3 let led = "";
4
5 if (smoke) {
6   led = "red"; // Αν υπάρχει καπνός, ανάβει κόκκινο ανεξαρ
7 } else if (temp < 20) {
8   led = "blue";
9 } else if (temp < 25) {
10  led = "green";
11 } else if (temp < 30) {
12  led = "yellow";
13 } else {
14   led = "red";
15 }
16
17 msg.payload = { led: led };
18 return msg;
19

```

Edit function node

Delete Cancel Done

⚙ Properties

📌 Name Καπνός → buzzer activation

⚙ Setup On Start On Message On Stop

```

1 let smoke = msg.payload.smoke;
2
3 msg.payload = { buzzer: smoke }; // true αν smoke==true
4 return msg;
5

```


Edit function node

Delete Cancel Done

Properties

Name Χρώμα LED (payload.led)

Setup On Start On Message On Stop

```
1 msg.payload = msg.payload.led;
2 return msg;
```

Edit function node

Delete Cancel Done

Properties

Name Εmoji Καπνός

Setup On Start On Message On Stop

```
1 msg.payload = msg.payload.smoke === true ? "Ανιχνεύτηκε καπνός" : "Καπνός απουσιάζει";
2 return msg;
```

Edit function node

Delete Cancel Done

Properties

Name Εmoji Θέρμανσης

Setup On Start On Message On Stop

```
1 msg.payload = msg.payload.heating === true ? "Θέρμανση ON 🔥" : "Θέρμανση OFF ❄️";
2 return msg;
```

Edit function node

Delete Cancel Done

Properties

Name Λήψη Θερμοκρασίας

Setup On Start On Message On Stop

```
1 msg.payload = msg.payload.temperature;
2 return msg;
```

Edit function node

Delete Cancel Done

Properties

Name Λήψη Υγρασίας

Setup On Start On Message On Stop

```

1 msg.payload = msg.payload.humidity;
2 return msg;

```

Edit function node

Delete Cancel Done

Properties

Name Θερμοστάτης Λογική

Setup On Start On Message On Stop

```

1 // Αν είναι επιθυμητή θερμοκρασία → αποθήκευση
2 if (msg.topic === "desired_temperature") {
3     flow.set("target_temp", msg.payload);
4     return null;
5 }
6
7 // Αν είναι θερμοκρασία περιβάλλοντος
8 if (msg.topic === "fire_status") {
9     let current = msg.payload.temperature;
10    let target = flow.get("target_temp") || 22;
11
12    msg.payload = { heating: current < target };
13    return msg;
14 }
15
16 return null;

```

Edit mqtt out node

Delete Cancel Done

Properties

Server 192.168.1.51:1883

Topic led_control

QoS 0 Retain

Name Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Edit mqtt out node

Delete Cancel Done

Properties

Server 192.168.1.51:1883

Topic buzzer_control

QoS 0 Retain

Name Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Edit mqtt out node

Delete Cancel Done

Properties

Server 192.168.1.51:1883

Topic heating_control

QoS 0 Retain

Name heating_control

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Delete Cancel Done

Properties

Group [Dashboard] Αισθητήρες

Size 6 x 5

Type Gauge

Label Θερμοκρασία

Value format {{msg.payload.temperature}}

Units °C

Range min 0 max 50

Colour gradient

Sectors 0 ... 20 ... 25 ... 50

Fill gauge from centre. ☐

Class Optional CSS class name(s) for widget

Name Θερμοκρασία

Edit gauge node

Delete Cancel Done

Properties

Group [Dashboard] Αισθητήρες

Size 6 x 5

Type Gauge

Label Υγρασία

Value format {{msg.payload.humidity}}

Units %

Range min 0 max 100

Colour gradient

Sectors 0 ... 30 ... 60 ... 100

Fill gauge from centre. ☐

Class Optional CSS class name(s) for widget

Name Υγρασία

Edit led node

Delete Cancel Done

Properties

Group [Dashboard] Αισθητήρες

Size 5 x 1

Label Ενεργό λαμπάκι LED

Label Placement left Label Alignment right

Shape circle

Show glow effect around LED ☒

Preview

Colors for value of msg.payload

msg.payload	Color
red	red
yellow	yellow
green	green
blue	blue

Edit text node

Delete Cancel Done

Properties

Group [Dashboard] Αισθητήρες

Size 6 x 1

Label

Value format {{msg.payload}}

Layout

label value label value label value

label value label value

Style ☐ Apply Style

Class Optional CSS class name(s) for widget

Name Καπνός Emoji Text

Edit text node

Delete Cancel Done

Properties

Group [Dashboard] Αισθητήρες

Size 6 x 1

Label

Value format {{msg.payload}}

Layout

label value label value label value

label value label value

Style ☐ Apply Style

Class Optional CSS class name(s) for widget

Name Θέρμανση Emoji Text

Edit chart node

Delete Cancel Done

Properties

Group [Dashboard] Γραφήματα

Size 6 x 5

Label Θερμοκρασία (°C)

Type Line chart ☐ enlarge points

X-axis last 20 minute: OR 1000 points

X-axis Label HH:mm:ss ☐ as UTC

Y-axis min 15 max 35

Legend None Interpolate linear

Series Colours

Blank label display this text before valid data arrives

Class Optional CSS class name(s) for widget

Name Θερμοκρασία

Edit chart node

Delete Cancel Done

Properties

Group [Dashboard] Γραφήματα

Size 6 x 5

Label Υγρασία (%)

Type Line chart ☐ enlarge points

X-axis last 20 minute: OR 1000 points

X-axis Label HH:mm:ss ☐ as UTC

Y-axis min 0 max 100

Legend None Interpolate linear

Series Colours

Blank label display this text before valid data arrives

Class Optional CSS class name(s) for widget

Name Υγρασία

Edit slider node

Delete Cancel Done

Properties

Group [Dashboard] Γραφήματα

Size 6 x 1

Label Επιλογή Θερμοκρασίας

Tooltip optional tooltip

Range min 18 max 30 step 0.2

Output continuously while sliding

If msg arrives on input, pass through to output: ☒

When changed, send:

Payload Current value

Topic desired_temperature

Class Optional CSS class name(s) for widget

Name Επιθυμητή Θερμοκρασία

Edit text node

Delete Cancel Done

Properties

Group [Dashboard] Γραφήματα

Size 6 x 1

Label Επιθυμητή

Value format {{msg.payload}} °C

Layout

label value label value label value

label value label value

Style ☐ Apply Style

Class Optional CSS class name(s) for widget

Name Εμφάνιση Επιθυμητής

7. Node-RED Dashboard

Η οπτικοποίηση των δεδομένων γίνεται μέσω του Node-RED Dashboard.

Όταν λαμβάνεται νέο MQTT μήνυμα (fire_status), όλα τα στοιχεία του dashboard ανανεώνονται άμεσα.

Ο χρήστης μπορεί να αλλάξει την επιθυμητή θερμοκρασία ανά πάσα στιγμή, και το σύστημα ανταποκρίνεται άμεσα με ενεργοποίηση/απενεργοποίηση θέρμανσης.

Βασικές Συνιστώσες του UI:

➤ Gauges

- Θερμοκρασία (°C)
- Υγρασία (%)
- Παρουσιάζουν την τρέχουσα τιμή με χρωματική ένδειξη ανά επίπεδο

➤ Charts

- Δείχνουν το ιστορικό των τελευταίων μετρήσεων θερμοκρασίας και υγρασίας

➤ Slider

- Ο χρήστης ορίζει την επιθυμητή θερμοκρασία
- Στη συνέχεια συγκρίνεται με την πραγματική θερμοκρασία ώστε να ενεργοποιηθεί ή όχι το ρελέ

➤ Text Nodes

- Εμφάνιση επιλεγμένης επιθυμητής θερμοκρασίας
- Ενδείξεις για καπνό ή κατάσταση θέρμανσης

➤ Emoji

- Ενισχύουν την εμπειρία του dashboard και κάνουν πιο κατανοητή την κατάσταση με μία ματιά

➤ LED Icons

- Προσομοιώνουν την ενεργοποίηση των LEDs στο Raspberry Pi
- Χρωματικά αντιστοιχούν στις θερμικές ζώνες:
 - Μπλε: <20°C
 - Πράσινο: 20–25°C
 - Κίτρινο: 25–30°C
 - Κόκκινο: >30°C

