

Objectif : Mettre en place un serveur http et récupérer les données envoyées par le client.

On utilisera le logiciel curl pour faire les requêtes http.

1 – Requête GET

⇒ Lancer Node-RED sur la Raspberry

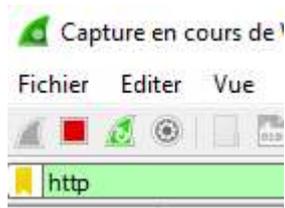
⇒ Placer les **nodes** "http in" et "http response" sur le plan, les configurer et les relier comme ci-dessous (pas de modification sur le node http response). **Ne pas oublier de déployer.**

⇒ indiquer /myserv pour l'url

The image displays the Node-RED interface for configuring an HTTP server. It is divided into four main sections:

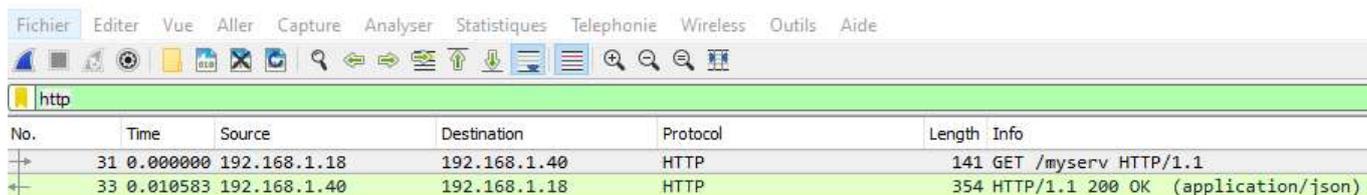
- Top-left:** A palette of nodes under the 'network' category. The 'http in' and 'http response' nodes are highlighted with a red circle.
- Top-right:** The 'Edit http in node' configuration panel. It includes a 'Delete' button and a 'Properties' section with the following settings:
 - Method: GET
 - URL: /myserv
 - Name: Name
- Bottom-left:** The 'Edit http response node' configuration panel. It includes a 'Delete' button and a 'Properties' section with the following settings:
 - Name: Name
 - Status code: msg.statusCode
 - Headers: (empty)
- Bottom-right:** A visual representation of the flow on a grid. It shows a '[get] /myserv' node connected to an 'http' node.

⇒ Lancer WireShark sur le PC et placer un filtre http



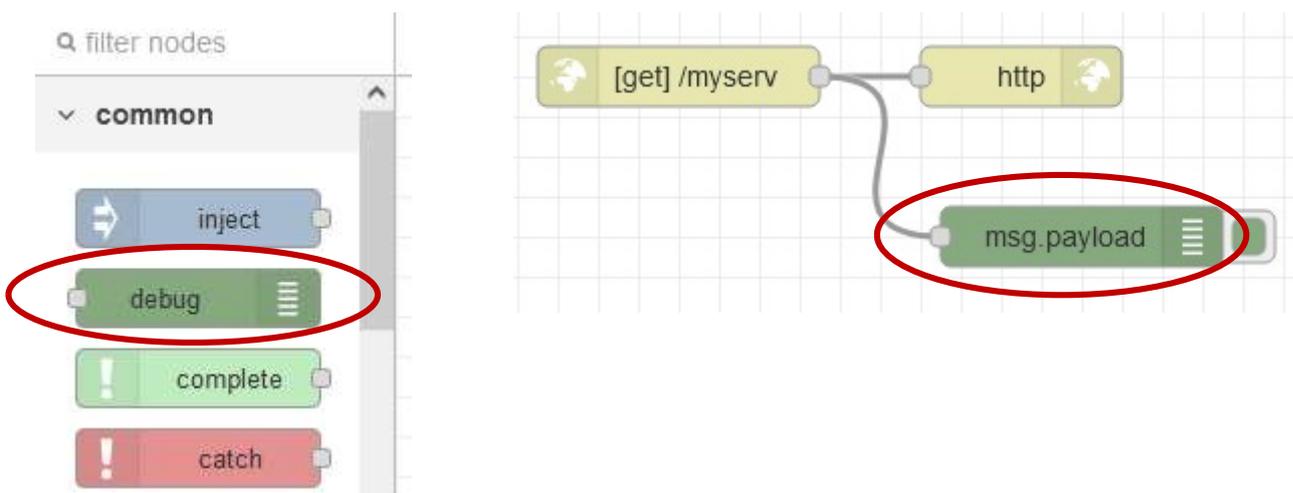
⇒ Sous l'invite de commande du PC, taper la commande suivante en remplaçant l'adresse IP de la raspberry :
curl "192.168.1.40:1880/myserv"

⇒ Vérifier les échanges sous WireShark comme ci-dessous :



No.	Time	Source	Destination	Protocol	Length	Info
31	0.000000	192.168.1.18	192.168.1.40	HTTP	141	GET /myserv HTTP/1.1
33	0.010583	192.168.1.40	192.168.1.18	HTTP	354	HTTP/1.1 200 OK (application/json)

⇒ Ajouter le node "debug" comme ci-dessous



⇒ Se placer en mode debug pour voir les messages, comme ci-dessous



⇒ Lancer la requête http ci-dessous (modifier l'IP Raspberry) et observer le message dans la fenêtre debug.

curl "192.168.1.40:1880/myserv?temp=15&humid=65"

2 – Requête POST

⇒ Modifier les 2 nodes précédents comme ci-dessous (méthode POST et ajout d'une entête type json en appuyant sur +add).

The image shows two screenshots of a REST client interface. The left screenshot is titled "Edit http in node" and shows a "Delete" button at the top. Below it is a "Properties" section with a "Method" dropdown set to "POST", an "Accept file uploads?" checkbox, a "URL" field containing "/myserv", and a "Name" field containing "Name". The right screenshot is titled "Edit http response node" and also has a "Delete" button. It features a "Properties" section with a "Name" field, a "Status code" field containing "msg.statusCode", and a "Headers" section. Under "Headers", there is a "Content-Type" dropdown set to "application/json". At the bottom of the right panel is a "+ add" button.

⇒ Sous l'invite de commande du PC, lancer notepad pour éditer et enregistrer le fichier capteur.json ci-dessous :

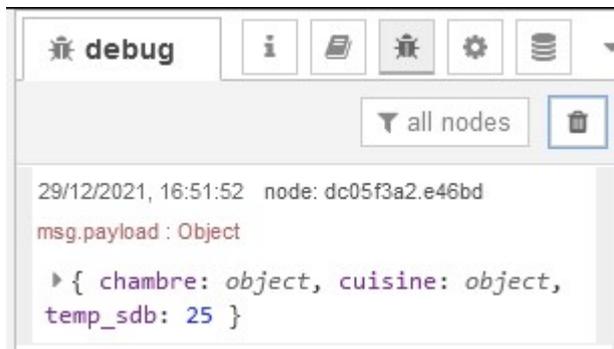
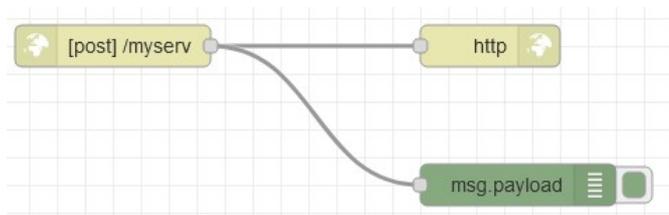
```
c:\users\dupont> notepad capteur.json
```

```
{
    "chambre":{
        "temperature":22,
        "id":"1234"
    },
    "cuisine":{
        "temperature":19,
        "id":"5678",
        "humidite":78
    },
    "temp_sdb":25
}
```

⇒ Lancer une requête http avec curl et envoi d'un fichier json (IP à modifier)

```
curl --data @capteur.json -H "Content-Type: application/json" "192.168.1.40:1880/myserv"
```

⇒ Observer les données dans la fenêtre debug



⇒ Ajouter une fonction et une jauge (échelle de 0 à 40) comme ci-dessous.

⇒ Relancer la même requête que précédemment et observer la donnée dans la fenêtre debug et dans l'interface utilisateur (192.168.1.40:1880/ui).

```
graph LR; A["[post] /myserv"] --> B["http"]; A --> C["function"]; B --> C; C --> D["gauge"]; C --> E["msg.payload"];
```

Edit function node

Delete

Properties

Name:

Setup | On Start | **On Message**

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

The image shows a Node-RED interface. On the left, the debug console is open, displaying a message from a node with ID 'cafe00ae.83f108' at 17:02:49 on 29/12/2021. The message payload is a number, '25'. On the right, a gauge widget titled 'Le groupe' is shown. The gauge has a scale from 0 to 40 units, with a needle pointing to the value 25. The gauge is partially filled with yellow, indicating the current value.

⇒ Modifier la fonction qui filtre temp_sdb par :

```
msg.payload=msg.payload.chambre.temperature;  
return msg;
```

⇒ Vérifier le nouvel affichage.

⇒ Ajouter une jauge qui affiche en plus l'humidité de la cuisine et faire vérifier au prof.