

SECRETARIA DE EDUCAÇÃO E CIÊNCIA  
INSTITUTO POLITÉCNICO DE BRAGANÇA  
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

LICENCIATURA EM ENGENHARIA DE INFORMÁTICA  
6º PERÍODO

FERNANDO SOUZA FURTADO CARRILHO  
JOSÉ RAFAEL SOARES BORGES

**RELATÓRIO PRÁTICO 04:**  
INTEGRAÇÃO DE DISPOSITIVOS IOT

BRAGANÇA  
2023

FERNANDO SOUZA FURTADO CARRILHO  
JOSÉ RAFAEL SOARES BORGES

**RELATÓRIO PRÁTICO 04:**  
INTEGRAÇÃO DE DISPOSITIVOS IOT

Este relatório objetiva a obtenção de nota na disciplina de Internet das Coisas dos graduandos no curso de Engenharia de Informática da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Bragança. Seu conteúdo é composto pela observação, descrição e aplicação referente à integração de dispositivos IoT.

## Sumário

<b>1</b>	<b>QUESTÃO 01: SERVIÇOS WEB</b>	<b>6</b>
1.1	LETRA [A]: DEFINIÇÃO DE SERVIÇOS WEB . . . . .	6
1.2	LETRA [B]: INTERFACES DE PROGRAMAÇÃO DE APLICAÇÕES . . . . .	6
1.3	LETRA [C]: API REST FULL . . . . .	7
<b>2</b>	<b>QUESTÃO 02: ASSISTENTE METEOROLÓGICO</b>	<b>8</b>
2.1	LETRA [B]: PREVISÃO METEOROLÓGICA DIÁRIA . . . . .	8
2.2	LETRA [D]: OBSERVAÇÃO METEOROLÓGICA DE ESTAÇÕES . . . . .	11
2.3	LETRA [E]: INTERFACE COM NODE-RED . . . . .	12
<b>3</b>	<b>QUESTÃO 03: ASSISTENTE VIRTUAL</b>	<b>16</b>
3.1	LETRA [C]: ASSISTENTE METEOROLÓGICO . . . . .	17

## Lista de Figuras

1	Fluxo Node-RED previsão de até 5 dias . . . . .	9
2	Configurações do HTTP Request no Node-RED . . . . .	9
3	Dados de saída do Debug no Node-RED da previsão de até 5 dias . . . . .	10
4	Dados de saída do Debug no Node-RED das últimas 24 horas . . . . .	11
5	Fluxo no Node-RED para as últimas 24 horas . . . . .	12
6	Fluxo no Node-RED para os próximos 5 dias . . . . .	12
7	Código 1 que trata a temperatura e humidade das últimas 24 horas . . . . .	13
8	Código 2 que trata a temperatura e humidade das últimas 24 horas . . . . .	14
9	Gráfico da Temperatura e Humidade das últimas 24 horas . . . . .	15
10	Código que trata a Temp. Min e Max e prob. de chuva dos próximos 5 dias . . . . .	15
11	Gráfico da Temp. Max, Min e da Prob. Chuva dos próximos 5 dias . . . . .	16
12	Composição do Assistente Virtual no Node-Red . . . . .	17
13	Primeira parte Assistente Virtual Node-Red . . . . .	18
14	Segunda Parte Assistente Virtual Node-Red . . . . .	19
15	Nó Switch . . . . .	20
16	Interface Node-Red Dashboard . . . . .	21
17	Fluxo para previsão de chuva hoje. . . . .	21
18	Fluxo para previsão de chuva amanhã. . . . .	21
19	Nó de função para previsão de chuva hoje. . . . .	22
20	Nó de função para a previsão de chuva amanhã. . . . .	23
21	Resultado previsão de chuva para hoje (03/05/2023). . . . .	23
22	Resultado previsão de chuva para amanhã (04/05/2023). . . . .	24
23	Fluxo para temperatura atual. . . . .	25
24	Fluxo para humidade atual. . . . .	25
25	Nó de função para filtro de valores da estação de Bragança . . . . .	26
26	Nó de função para mensagem de temperatura atual. . . . .	27
27	Nó de função para mensagem de humidade atual. . . . .	27
28	Nó de função para mensagem de temperatura atual. . . . .	28

29	Nó de função para mensagem de humidade atual. . . . .	29
----	---	----

## 1 QUESTÃO 01: SERVIÇOS WEB

Para este tópico, sua finalidade dá-se à definição de Serviços Web, Interfaces de Programação de Aplicações e sobre API Rest Full. À vista disso, seu intuito é dar abordagem de interpretação dos conceitos principais para atuação nos tópicos seguintes deste relatório.

### 1.1 LETRA [A]: DEFINIÇÃO DE SERVIÇOS WEB

De acordo com o W3 (W3, 2023), os *Serviços WEB* são uma grama de soluções usadas nas integrações de sistemas e na comunicação entre aplicações as distintas; também entendida como um conjunto de definições e protocolos para construir e integrar software de aplicação

Diante disso, ainda para a W3, para o o conceito de serviço, é possível afirmar que os serviços são providos de semântica que pode ser identificada em uma descrição de serviço e que pode ser expressa em uma linguagem de descrição de serviço.

Por fim, afirma mais ainda que, essa identificação da semântica de um serviço e, com o intuito para agentes mais avançados, com foco na descrição do próprio contrato de serviço, é permitido que os agentes que implementam serviços da Web determinem as capacidades de outros agentes pares (peer-to-peer).

### 1.2 LETRA [B]: INTERFACES DE PROGRAMAÇÃO DE APLICAÇÕES

Conforme expressa a Amazon Web Service, AWS (Amazon AWS, 2023), as *Interfaces de Programação de Aplicações*, do inglês *Application Programming Interface*, refere-se a qualquer software com uma função distinta. Dentre estas APIs está a mais popular, a exemplo, *API REST* com seus Verbos HTTP (os mais conhecidos):

1. POST - posta/insere dados em um sistema por intermédio de uma API;
2. GET - busca um ou mais dados, com ou sem filtros, em uma API;;
3. PUT - altera/atualiza as informações de um ou mais elementos dentro de uma API, de acordo com um identificador;

4. DELETE - deleta/apaga os dados de um ou mais elementos de uma API de acordo com um identificador;

Dessa forma, para a *AWS*, a interface pode ser pensada como um contrato de serviço entre duas aplicações. Consoante, é com base nesse contrato em que se define como as duas se comunicam usando solicitações e respostas.

Por fim, é baseado na documentação de suas respectivas APIs, em que se é armazenado as informações sobre como os desenvolvedores devem estruturar essas solicitações e respostas.

### 1.3 LETRA [C]: API REST FULL

Para o site ReadHat (ReadHat, 2023), uma API REST, também conhecida como *API RESTfull*, é uma interface de programação de aplicativos - API ou API da web - que está em conformidade com as restrições do estilo arquitetônico REST e permite a interação com os serviços da web RESTfull.

Enquanto isso, por outro lado, a sigla REST, vem do inglês, **RE**presentational **State** Transfer, em português, “**Transferência de Estado Representacional**” criado pelo cientista da computação Roy Fielding.

Diante disso, a sua concepção foi tido ”como uma abstração da arquitetura da web, a qual trata-se de um conjunto de princípios e definições necessários para a criação de um projeto com interfaces bem definidas” (TOTVS, 2020, 2020).

Por conseguinte, ainda para o RedHat, para que uma API seja considerada RESTful, ela deve atender a estes critérios:

- Arquitetura client-server: composta por clientes, servidores e seus recursos, com solicitações ordenadas por intermédio do HTTP;
- Comunicação cliente-server: composta por clientes e servidores sem estado. Isso implica que nenhuma informação do cliente é armazenada entre as solicitações GET e cada solicitação é separada e desconectada;

- Dados armazenados em cache: implica que existe dados armazenados em cache entre as interações do cliente e servidor;
- Interface uniforme: provida de uma interface entre os componentes para que as informações sejam transferidas, uniforme a qual permite que exista um formulário padrão.
- Sistema em camadas: permite a organização em cada tipo de servidor - responsáveis por: segurança, balanceamento de carga e outros - envolvido na recuperação das informações solicitadas em hierarquias, não visível aos clientes.
- Code-on-demand: refere-se à capacidade de enviar código executável do servidor para o cliente, no momento em que é requerido, o que amplia as funcionalidades do cliente. OBS.: Este é opcional.

Por fim, o REST é compreendido como um conjunto de diretrizes que podem ser implementadas conforme se tornar necessário, o que faz com que as APIs REST comportem-se de maneira mais rápidas, leves e com maior escalabilidade - ótima opção para Internet das Coisas (IoT) e desenvolvimento Mobile - .

## 2 QUESTÃO 02: ASSISTENTE METEOROLÓGICO

Para este tópico, seu intuito é a visualização do percurso para se desenvolver uma aplicação que opera como assistente meteorológico, considerando os serviços externos listados na página web do Instituto Português do Mar e da Atmosfera – IPMA, [acesse aqui](#).

Diante disso, para o estudo de caso, a cidade/local Bragança será a referência base, a qual há duas estações numeradas/identificadas em 1200575 e 1200576 as quais coletam os dados meteorológicos da cidade.

### 2.1 LETRA [B]: PREVISÃO METEOROLÓGICA DIÁRIA

Para esta atividade objetiva-se com o uso do nó *"http request"*, o qual é disponível no Node-RED e *"globalIdLocal"*, explicar quais informações são possíveis obter utilizando o



serviço *"Previsão Meteorológica Diária até 5 dias agregada por Local"*.

Para tanto, dito isso, no Node-RED faz-se o fluxo para que seja acessado a API para capturar os dados REST da previsão meteorológica, de até 5 dias. Diante disso, para visualizar o fluxo realizado, a Figura ?? a seguir expressa-o.

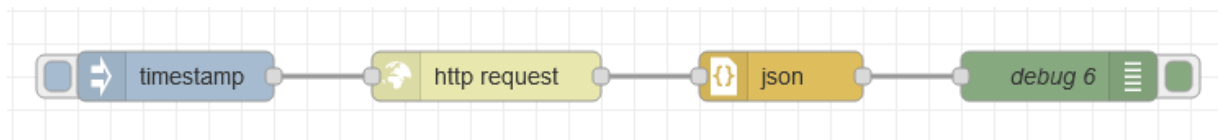


Figura 1: Fluxo Node-RED previsão de até 5 dias

Em sequência, da Figura 1, é possível ver que há 4 nós no Node-RED. Consoante, da esquerda para a direita, há os seguintes nós nomeados em, respectivamente: *timestamp*, *http request*, *json* e *debug 6*.

À vista disso, o nó *timestamp* foi adicionado para que a partir dele seja iniciado a ação para que o *http request* seja acionado. Dessa forma, no nó do *http request* foi adicionado o endereço do link <https://api.ipma.pt/open-data/forecast/meteorology/cities/daily/globalIdLocal.json>, de tal forma que o *globalIdLocal* seja substituído por 1010500.

Consoante, o link final fica em <https://api.ipma.pt/open-data/forecast/meteorology/cities/daily/1010500.json>, de tal forma que a solicitação REST seja *GET*, conforma Figura 2 expressa.



Figura 2: Configurações do HTTP Request no Node-RED

Dessa forma, feito a configuração no nó do *http request*, o nó próximo *json* dita que a resposta do *http request* seja passado em *JSON Object* para o nó *Debug 6*, o qual apresenta

os dados no Debug do Node-RED.

Com vista à isso a Figura 3 expressa o resultado da solicitação iniciada no *timestamp* e apresentado pelo *Debug 6*.



Figura 3: Dados de saída do Debug no Node-RED da previsão de até 5 dias

Por fim, conforme é visível na Figura 3, acima apresentada, há diversos dados que foram buscados pelo nó do *http request*, dentre eles estão: o órgão "IPMA", o país PT, Portugal, os dados (a precipitação, a temperatura máxima, temperatura mínima, longitude e outros), a data de acesso e outros.

## 2.2 LETRA [D]: OBSERVAÇÃO METEOROLÓGICA DE ESTAÇÕES

Por conseguinte, semelhante ao exercício anterior, o objetivo deste tópico é utilizar o nó *"http request"*, mas a diferença está em explicar quais informações são possíveis obter utilizando o serviço *"Observação Meteorológica de Estações (dados horários, últimas 24 horas)"*.

Diante disso, para que seja possível fazer essa atividade, o fluxo de processo é o mesmo que o da Figura 1, entretanto, a única diferença é o link de inserção no *http request*, o qual é link [https://api.ipma.pt/open-data/observation/meteorology/stations/ observations.json](https://api.ipma.pt/open-data/observation/meteorology/stations/observations.json) , no campo de URL da Figura 2.

Em seguida, o nó de *Debug 6* da Figura 1, com as alterações acima citadas, ao acionar o *timestamp* ocorre a seguinte saída, conforme expressa a Figura 4.

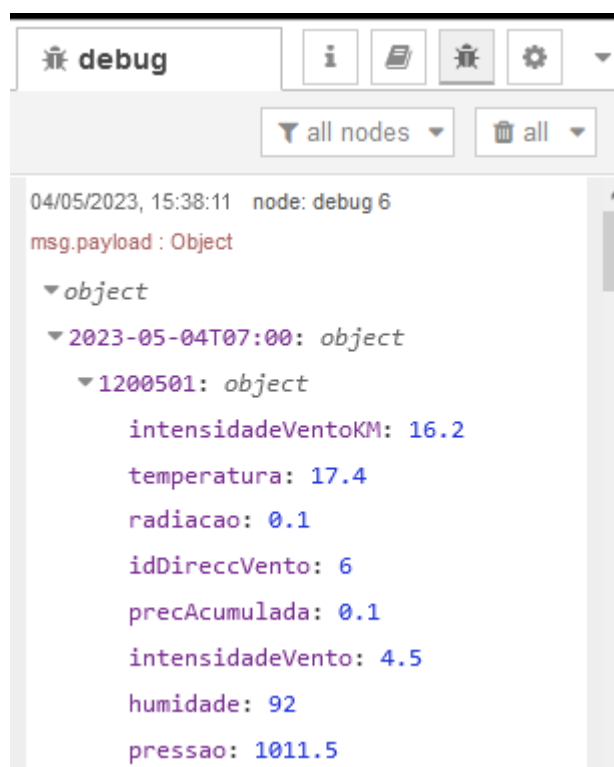


Figura 4: Dados de saída do Debug no Node-RED das últimas 24 horas

Por fim, conforme é visível na Figura 4, acima apresentada, a saída da busca no endereço inserido no nó *http request* estão, respectivamente: (1) data e horários das últimas 24 horas; (2) em cada data estão todas as estações existentes; (3) dentro de

cada estação estão os dados capturados pelos sensores na estação, tais como: pressão, temperatura, humidades e outros.

### 2.3 LETRA [E]: INTERFACE COM NODE-RED

Para esta atividade, seu intuito é desenvolver uma interface com o Node-RED que apresente num gráfico a temperatura e humidade disponível na estação meteorológica de Bragança nas últimas 24 horas. Consoante, pede-se também, a adição na interface a probabilidade de chuva, temperatura máxima e mínima para os próximos 5 dias.

Isto posto, para a atuação com foco na resolução das metas acima apresentadas, a seguir está a Figura 5, a qual elucida o fluxo que representa o trâmite e tratamento dos dados para a captação dos dados das últimas 24 horas e posteriormente apresentar o gráfico de temperatura e humidade.

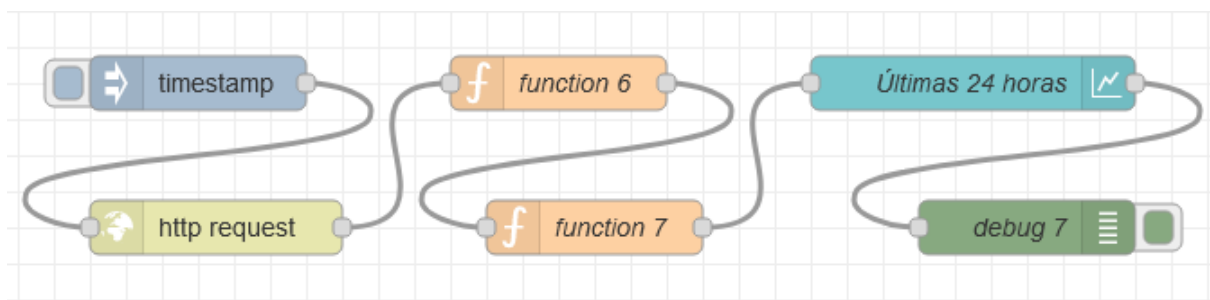


Figura 5: Fluxo no Node-RED para as últimas 24 horas

Diante disso, apresentado o fluxo no Node-RED para as últimas 24 horas, em Bragança-PT, na Figura 5 acima, a seguir consta a Figura 6, a qual expressa o fluxo no Node-RED para os próximos 5 dias, com valores da Temperatura Máxima, Mínima e a Probabilidade de Chuva, ainda na cidade de Bragança-PT.

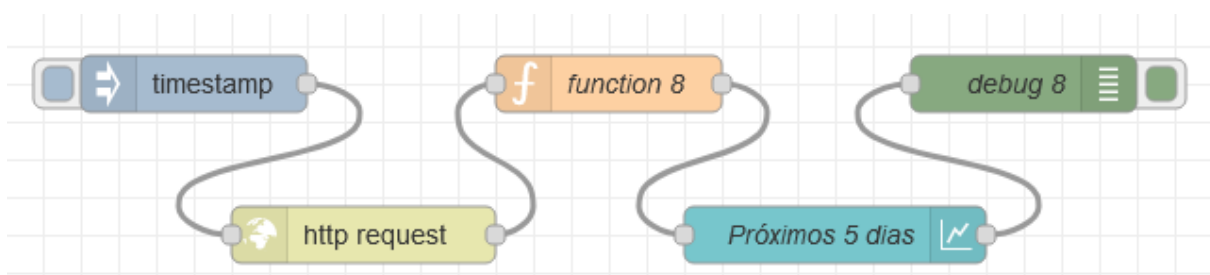


Figura 6: Fluxo no Node-RED para os próximos 5 dias

Para a Figura 5 o link utilizado para acessar as últimas 24 horas foi o mesmo utilizado para chegar à Figura 4, enquanto o link para acessar os próximos 5 dias foi o mesmo link usado na Figura 2.

Dito isso, em seguida, no nó *function 6* da Figura 5, a Figura 7 abaixo, expressa o código no qual faz o tratamento de: ordenar as datas e horários, decrescentemente, da estação meteorológica de Bragança, com intuito de ter as 24 últimas horas e enviar; dos valores da temperatura e humidade.

```
1  var dados = msg.payload;
2  var data = Object.keys(msg.payload);
3  var dateString = Object.keys(msg.payload);
4  var dateSort = [];
5
6  // @ts-ignore
7  dateSort = dateString.sort((date1, date2) => new Date(date2) - new Date(date1));
8
9  var agora = new Date().getHours()-1;
10
11  var horas = {}
12  for (let i = 0; i < 24; i++) {
13    agora = (agora < 0 ? 23 : agora);
14    horas["h_" + agora.toString()] = {
15      "st1": {
16        "temperatura": msg.payload[Object.keys(msg.payload)[data.indexOf(dateSort[i])]]
17        ['1200575']['temperatura'],
18        "humidade": msg.payload[Object.keys(msg.payload)[data.indexOf(dateSort[i])]]
19        ['1200575']['humidade']
20      }
21    }
22  }
23  agora--;
24 }
25 horas['sort'] = dateSort;
26 msg.payload = horas;
27 return msg;
```

Figura 7: Código 1 que trata a temperatura e humidade das últimas 24 horas

Diante disso, veja na Figura 7, que na linha 16 e 19, respectivamente, que no objeto criado na linha 11, tratado na linha 14, estabelece os valores de temperatura e humidade para a estação de número 1200575.

Paralelamente, perceba que, ainda na Figura 7, na linha 7, as data recebidas pelo nó *http request*, são ordenadas de maneira decrescente, uma vez que o desejado são as últimas 24 horas.

Com vista à isso, repare que na linha 26 o *msg.payload* recebe o *JSON Object* para

que na linha seguinte ocorra o retorno do *msg*.

Por conseguinte, a seguir, na Figura 8, encontra-se o segundo tratamento dos dados capturados pelo *http request*, da Figura 5, pre-tratados na Figura 7, com intuito de mandar ao *chart*, nomeado em **últimas 24 horas**, para que seja possível ter visualização em gráfico.

```
1  var temp = []; var hum = [];
2
3  for (let x = 0; x < 24; x++){
4      temp[x] = msg.payload[Object.keys(msg.payload)[x]]['st1']['temperatura'];
5      hum[x] = msg.payload[Object.keys(msg.payload)[x]]['st1']['humidade'];
6  }
7  var chart = [{
8      "labels":msg.payload['sort'],
9      "data": [temp, hum],
10     "series": ["Temperatura","Humidade"]
11 }];
12 msg.payload = chart;
13 return msg;
```

Figura 8: Código 2 que trata a temperatura e humidade das últimas 24 horas

Isto posto, conforme é visível na Figura 8, está presente o código que trata os dados recebidos do código da Figura 7 e os envia para o nó Chart, nomeado em **últimas 24 horas**.

Com vista à isso, repare que como é visto na Figura 8, a estrutura que é enviada ao Chart, é um *JSON Object* no qual há três atributos, os quais são: **labels**, **data** e **series**. Dessa forma, note que o **labels** é atributo no qual se armazena os elementos de coluna, o qual expressa cada elemento das abscissas.

Por conseguinte, já o **data** é o atributo no qual se adiciona os dados desejados, enquanto o **series** é o local no qual se inclui os elementos de legendas de cada elemento de atributo incluso em **data**.

Nesse sentido, ao término das configurações do código 2, Figura 8, os valores são submetidos ao Chart, o qual imprime um gráfico com os valores e parâmetros apresentados. Para tanto, a figura seguinte, Figura 9, expressa os dados que compõem os valores recebidos do *http request* da Figura 5, em formato visual, gráfico.

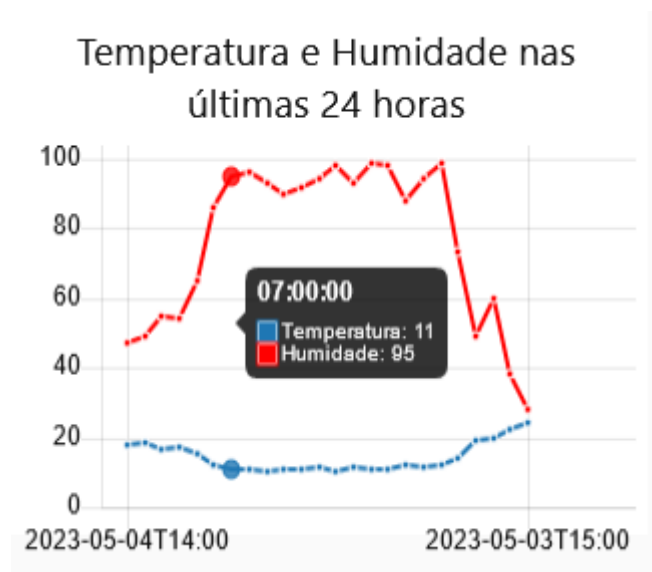


Figura 9: Gráfico da Temperatura e Humidade das últimas 24 horas

Nesta vista, é possível ver na Figura 9 que os valores da *Temperatura* e *Humidade* foram encontrados, na estação de Bragança, de número 1200575. Logo, conforme é visto na figura acima, às 07h AM a temperatura esteve a 11 °C, quanto a humidade esteve a 95 %.

Em seguida, com foco no processo de se chegar ao gráfico do fluxo da Figura 6, o passo inicial é ver que com ativar o *timestamp*, o *http request* capturará os dados fornecidos pela API Rest. À vista disso, o passo seguinte é usar o nó *function 8* para tratar os dados e enviar ao *Chart*, com o código abaixo da Figura 10, abaixo.

```

1  var tMin = []; var tMax = []; var pChu = [];
2
3  for (let x = 0; x < 5; x++) {
4      tMin[x] = msg.payload['data'][x]['tMin'];
5      tMax[x] = msg.payload['data'][x]['tMax'];
6      pChu[x] = msg.payload['data'][x]['precipitaProb'];
7  }
8
9  var chart = [{
10     "labels": ["Dia 1", "Dia 2", "Dia 3", "Dia 4", "Dia 5"],
11     "data": [tMin, tMax, pChu],
12     "series": ["Temperatura Mínima", "Temperatura Máxima", "Probabilidade de Chuva"]
13  }];
14
15  msg.payload = chart;
16  return msg;

```

Figura 10: Código que trata a Temp. Min e Max e prob. de chuva dos próximos 5 dias

Em continuidade, conforme é possível ver na Figura 10, os valores das temperaturas mínimo, máxima e da probabilidade de chuvas é inserida no campo da **data** do corpo do *chart*.

Já o campo **labels**, recebe a lista dos nomes dos dias da semana. Por outro lado, o campo **series**, recebe o nome de: *Temperatura Mínima*, *Temperatura Máxima* e *Probabilidade de Chuva*. Isso feito, os valores tratados são enviados ao *Chart* o qual permite ter o gráfico conforme é visível abaixo na Figura 11.

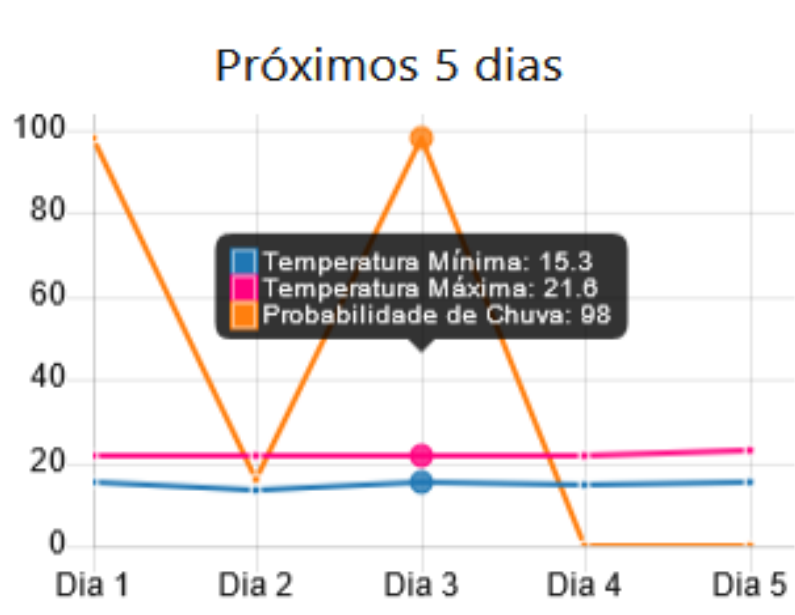


Figura 11: Gráfico da Temp. Max, Min e da Prob. Chuva dos próximos 5 dias

Ademais, de acordo como se vê na Figura 11, os valores da Temperatura Mínima, Máxima e da Probabilidade de Chuvas estão expressas nos próximos 5 dias, tido a data atual de aplicação.

Por fim, vê-se que com as devidas aplicações, é possível aplicar com Node-RED e com usufruto da aplicação IPMA, encontrar informações válidas e verídicas e maneira fácil, simplificada e eficiente.

### 3 QUESTÃO 03: ASSISTENTE VIRTUAL

Este tópico, trata-se da aplicação e atuação de um assistente virtual para informações meteorológicas. Em seu escopo, encontra-se de maneira única, a demonstração e usabili-



dade de um assistente meteorológico.

### 3.1 LETRA [C]: ASSISTENTE METEOROLÓGICO

É proposto desenvolver um assistente virtual utilizando o Node-RED e a API do IPMA para fornecer informações meteorológicas em resposta a comandos de voz do usuário.

O assistente deve ser capaz de interpretar e responder a quatro tipos de comandos: previsão de chuva hoje, previsão de chuva amanhã, temperatura atual e humidade atual.

Para obter as informações solicitadas, utilizaremos dois serviços diferentes da API IPMA:

- O serviço "Previsão Meteorológica Diária até 5 dias agregada por Local" para as previsões de chuva;
- O serviço "Observação Meteorológica de Estações" para as informações de temperatura e humidade atual.

O Node-RED será responsável por interpretar os comandos de voz do usuário, chamar os serviços apropriados da API IPMA e fornecer as respostas para cada um dos casos. Na imagem abaixo, figura 12 é ilustrada a vista geral do assistente virtual:

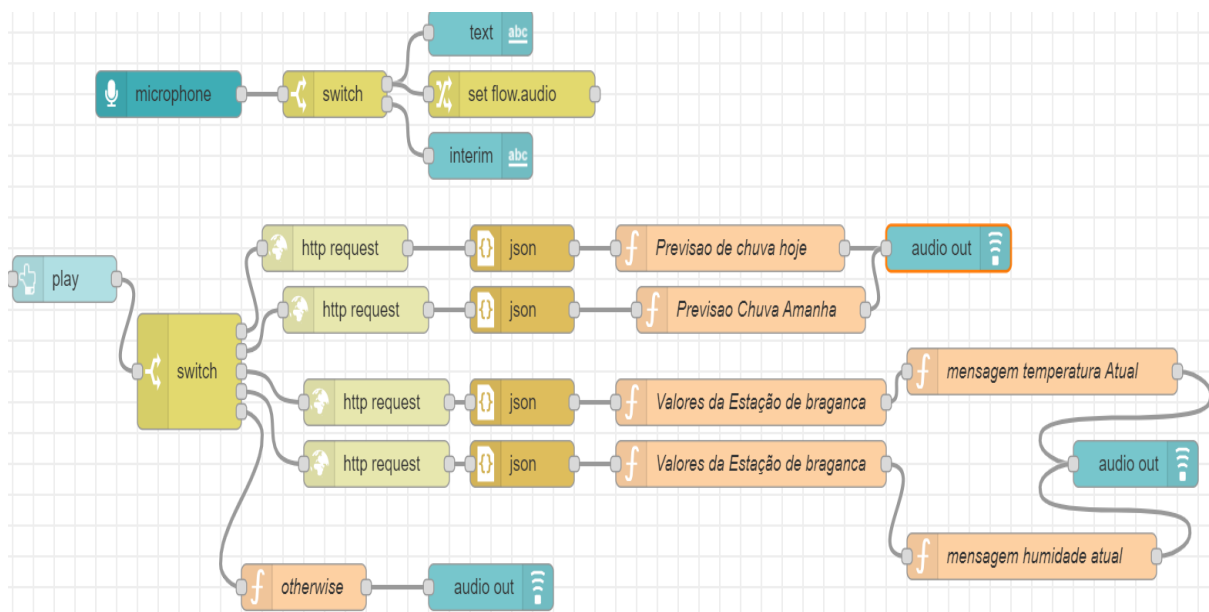


Figura 12: Composição do Assistente Virtual no Node-Red

Nesta primeira parte, Figura 13, é configurado o nó de microfone (node-red-node-ui-microphone) no modo de Speech Recognition, permitindo que o utilizador fale em vez de digitar os comandos no Node-RED.

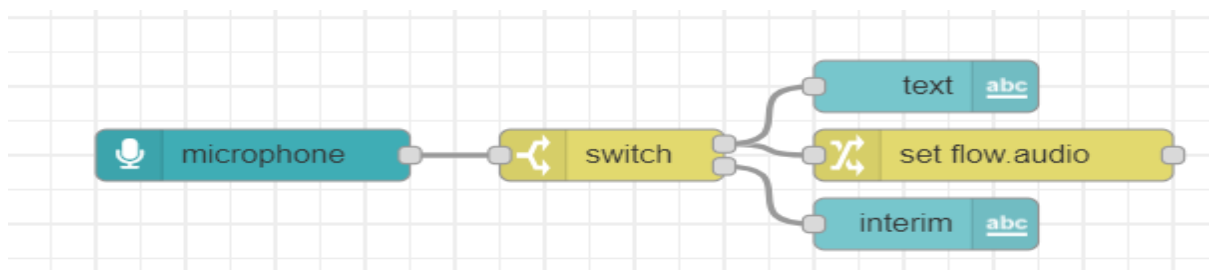


Figura 13: Primeira parte Assistente Virtual Node-Red

Ele usa um serviço de reconhecimento de fala para converter o áudio capturado pelo microfone em texto e enviar a mensagem de texto resultante para o fluxo do Node-RED.

Quando o nó de microfone captura o áudio, ele envia a mensagem com `msg.done` igual a `false`, o que indica que o processo de reconhecimento de fala ainda está em andamento. Quando o áudio é completado e o texto é convertido, o nó de microfone envia uma mensagem com `msg.done` igual a `true`, indicando que o processo de reconhecimento do áudio foi concluído com sucesso.

O nó Switch é usado para testar o valor de `msg.done` e encaminhar a mensagem para o fluxo correto com base nesse valor. Se `msg.done` for `false`, o fluxo encaminha a mensagem para o nó `interim` para exibir o texto parcialmente reconhecido em tempo real. Se `msg.done` for `true`, o fluxo encaminha a mensagem para o nó `text` para exibir o texto completo do reconhecimento de fala.

No nó de `change`, variável `"flow.audio"` é usada no fluxo como uma variável global para armazenar o áudio capturado de forma que possa ser acessado posteriormente em outros nós ou fluxos.

Mas para que o áudio chegue até a variável `"flow.audio"`, ele precisa ser armazenado na propriedade `"payload"` do objeto `"msg"` em algum momento do fluxo. Isso pode ser feito, por exemplo, com o uso de um nó `"change"` para definir o valor da propriedade `"audio"` do objeto `"flow"` como sendo igual ao valor da propriedade `"payload"` do objeto `"msg"`.

Na segunda parte da estrutura do assistente virtual, figura 14 é definido o tratamento para o texto reconhecido.

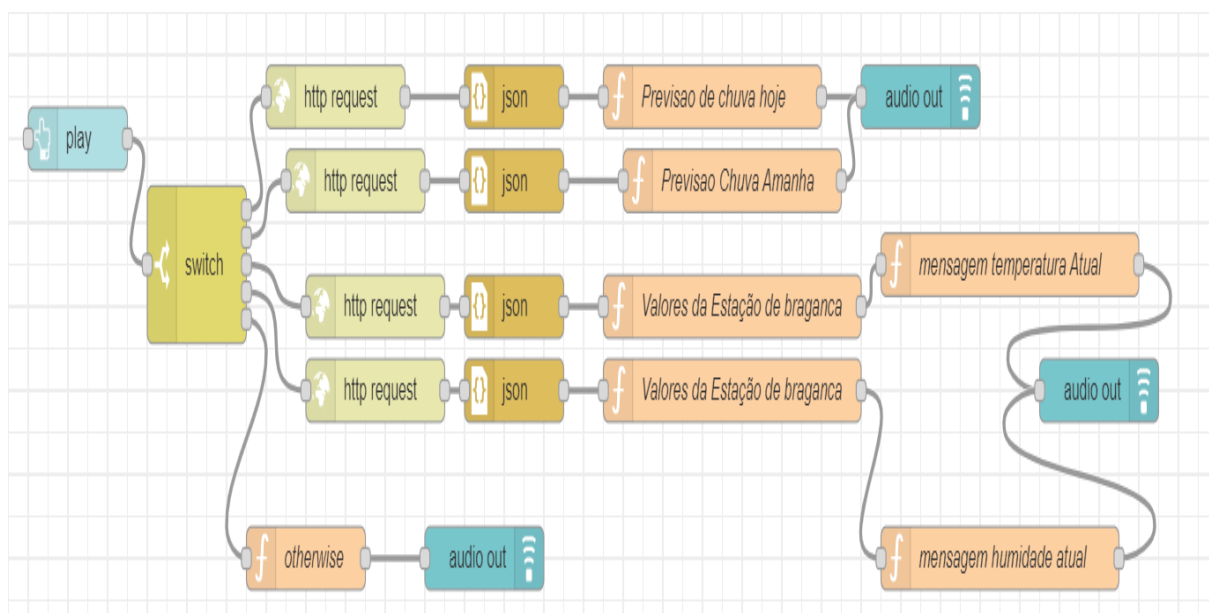


Figura 14: Segunda Parte Assistente Virtual Node-Red

Para isso, começa-se com o nó de botão, uma forma de gerar uma mensagem de saída quando um botão é pressionado na interface do Node-RED. Neste caso, o valor da variável global, `flow.audio` é atribuído ao `msg.payload`, o que quer dizer que quando o botão é pressionado, o fluxo de saída é enviado com o conteúdo da variável global `flow.audio` como `msg.payload`.

O nó de switch, figura 15, é usado para avaliar o valor de uma mensagem e encaminhá-la para diferentes saídas com base em seu conteúdo.



Figura 15: Nó Switch

No caso deste fluxo, a mensagem que está sendo avaliada é o texto reconhecido pelo microfone do usuário. O switch irá comparar o valor da mensagem com várias opções de entrada e, dependendo do valor da mensagem, encaminhará a mensagem para uma saída específica, como a exemplifica a figura 15 acima.

De seguida, para cada opção é usado o nó "HTTP Request" para fazer uma solicitação à API do IPMA para obter informações sobre a previsão meteorológica, usando o formato JSON, a partir do nó Json, presente em cada percurso.

No Node-Red dashboard, figura 16, o utilizador consegue gravar áudio com o microfone, que é convertido em texto usando a tecnologia de reconhecimento de fala. O texto é então processado e uma resposta de áudio é gerada.

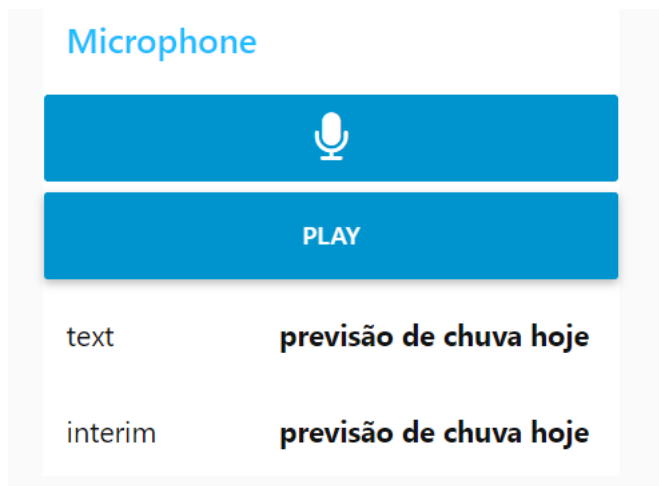


Figura 16: Interface Node-Red Dashboard

Dessa forma, quando se interage com o nó de botão, identificado como play na figura 16, a resposta de áudio é reproduzida no nó Audio Out, permitindo que o usuário ouça a resposta.

### Ponto 1 e 2, Previsão de Chuva Hoje e Previsão de chuva amanhã

Estas 2 questões foram colocadas juntas, porque a única diferença entre elas, na forma de construção da alínea, é um parâmetro do array retornado pela API do IPMA. Fica a demonstração para os 2 casos.

- Fluxo Node-Red, previsão de chuva para hoje :



Figura 17: Fluxo para previsão de chuva hoje.

- Fluxo Node-Red, previsão de chuva para amanhã :



Figura 18: Fluxo para previsão de chuva amanhã.

Depois de a mensagem chegar ao switch, é encaminhada para o respetivo caminho. Com isto é realizado o pedido à API do IPMA, ele retorna o objeto com as informações relevantes para Previsão Meteorológica Diária até 5 dias agregada por Local, usando o seguinte endereço: <https://api.ipma.pt/open-data/forecast/meteorology/cities/daily/1040200.json>, de notar de inserir o globalIDLocal de Bragança, igual a 1040200.

A informação retornada relevante, está presente em um array de dados de tamanho igual a 5, que devolve a previsão meteorológica para os próximos 5 dias a contar do dia atual. Portanto, a posição zero [0] refere-se ao dia de hoje, posição [1] ao dia de amanhã, e assim por diante até à posição n-1.

De seguida, é usado o nó de função, usado para compor uma mensagem informativa com a previsão para o dia de hoje a ser enviada para o nó de audio out, usado para reproduzir áudio no navegador, foi usado também um nó de debug para mostrar a mensagem na consola.

Para a previsão de chuva hoje, foi usada a seguinte função:

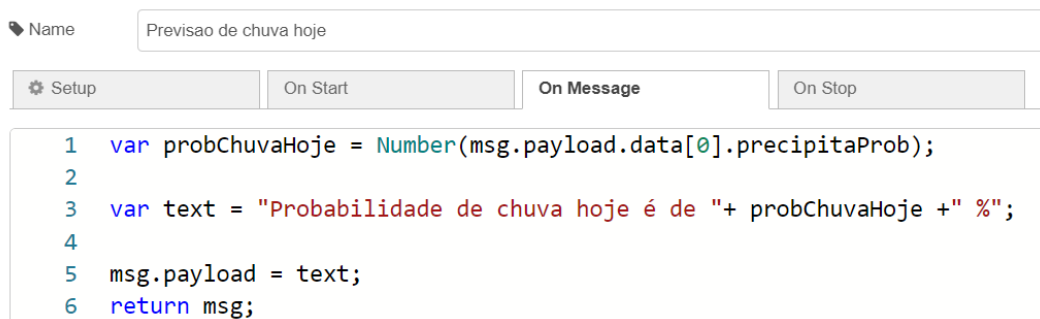


Figura 19: Nó de função para previsão de chuva hoje.

De igual, forma para a previsão de chuva de amanhã, foi utilizada em baixo refre-sentada, mudando apenas a posição do array que passa de 0 (referência ao dia de hoje) para 1 (referência do dia de amanhã).

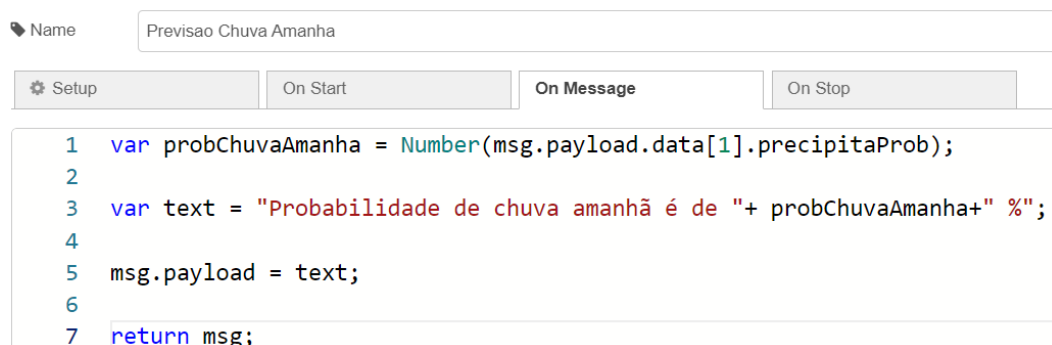


Figura 20: Nó de função para a previsão de chuva amanhã.

Por fim, nas próximas figuras, é dado o resultado final esperado destes 2 fluxos, e uma vez que é impossível reproduzir o áudio inerente à pergunta neste documento, juntou-se um nó de debug, para visualizar o esperado.

Para o fluxo, “Previsão de chuva hoje”, o atributo de probabilidade de precipitação (precipitaProb) é igual a 100 para o dia (03/05/2023), portanto, a mensagem gerada será “Probabilidade de chuva hoje é de 100%”, como consta a seguinte figura 21:

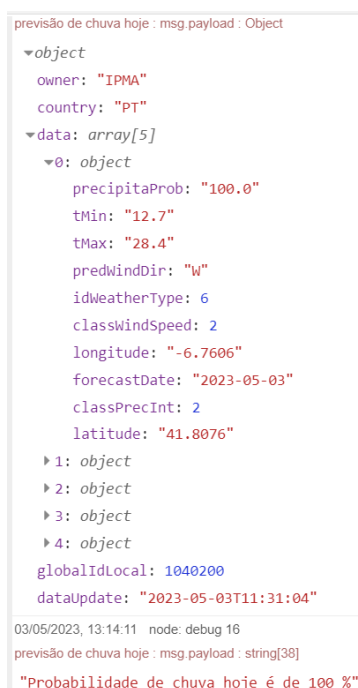


Figura 21: Resultado previsão de chuva para hoje (03/05/2023).

Para os valores de humidade, o último valor é igual a 53

```
previsão de chuva amanhã : msg.payload : Object
▼ object
  owner: "IPMA"
  country: "PT"
  ▼ data: array[5]
    ▶ 0: object
    ▼ 1: object
      precipitaProb: "91.0"
      tMin: "10.8"
      tMax: "20.6"
      predWindDir: "SW"
      idWeatherType: 6
      classWindSpeed: 2
      longitude: "-6.7606"
      forecastDate: "2023-05-04"
      classPrecInt: 2
      latitude: "41.8076"
    ▶ 2: object
    ▶ 3: object
    ▶ 4: object
  globalIdLocal: 1040200
  dataUpdate: "2023-05-03T11:31:04"
03/05/2023, 13:16:09 node: debug 16
previsão de chuva amanhã : msg.payload : string[39]
"Probabilidade de chuva amanhã é de 91 %"
```

Figura 22: Resultado previsão de chuva para amanhã (04/05/2023).



### Ponto 3 e 4, Temperatura e Humidade atual

Para estas 2 questões, a sua maneira de execução é muito semelhante, mudando apenas de valor qualitativo, enquanto um fluxo analisa o valor de temperatura atual, o outro explora o valor de humidade atual.

- Sequência no Node-Red para Temperatura Atual:



Figura 23: Fluxo para temperatura atual.

- Sequência no Node-Red para Humidade Atual:

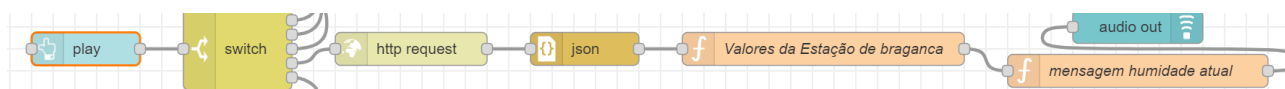


Figura 24: Fluxo para humidade atual.

Depois de o áudio do utilizador destes 2 pontos ser reconhecido e passar pelo nó de switch, para ambos os fluxos é realizado um pedido HTTP à API do IPMA para o serviço "Observação Meteorológica de Estações (dados horários, últimas 24 horas)".

Formatando a resposta em formato JSON, os dados retornados, chegam em valores de data hora, das últimas 24, para todas as estações de Portugal, sendo necessário filtrar este resultado para apenas valores da estação de Bragança, alvo de estudo, identificada com o número 1200575.

Assim sendo, é utilizado o mesmo nó de função, em cada um dos fluxos acima representados, para a filtragem de dados para a estação de Bragança, como mostra a figura:

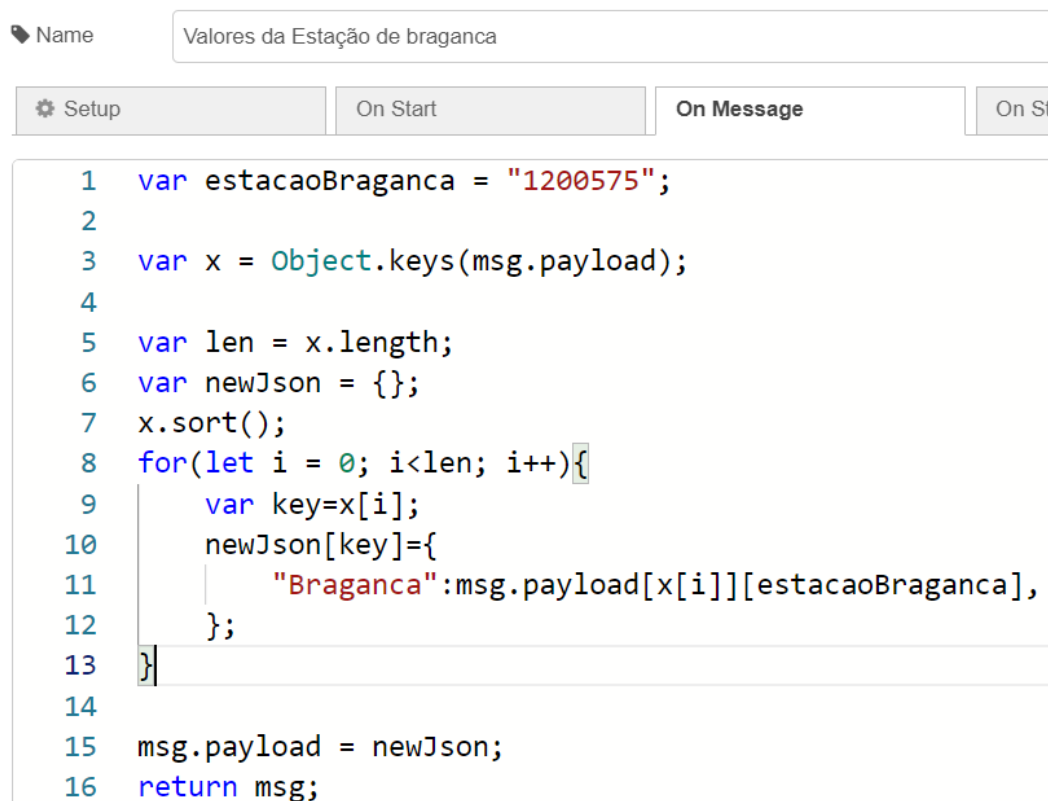


Figura 25: Nó de função para filtro de valores da estação de Bragança

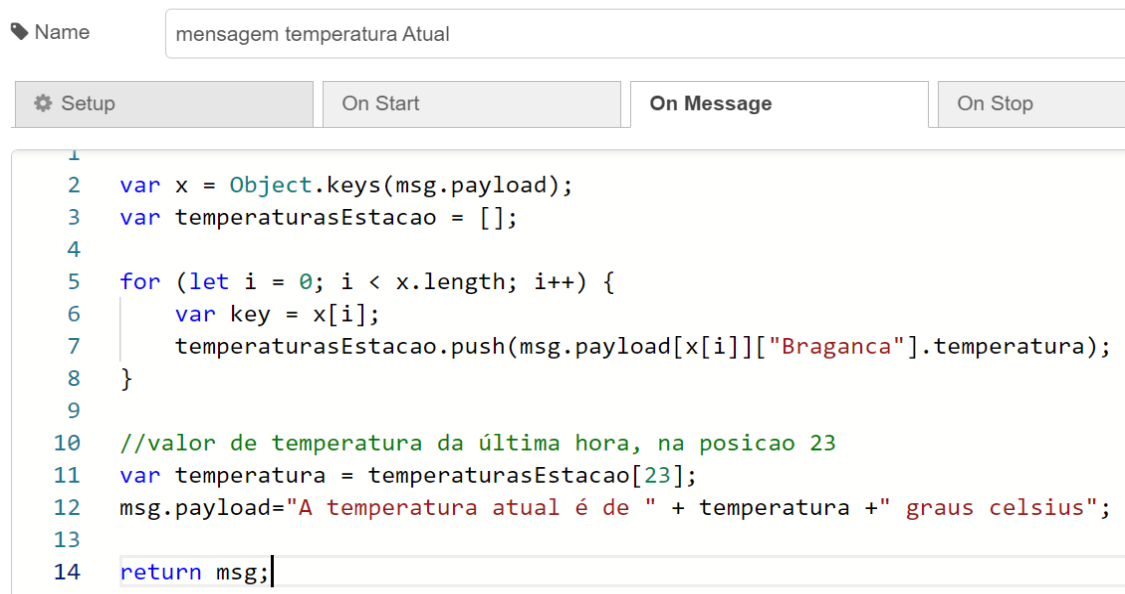
Primeiramente, é definida a variável "estacaoBraganca" com o código da estação de Bragança. Em seguida, é criada uma variável "x" que recebe as chaves (keys) do objeto JSON original que contém os dados da API. A variável "len" recebe o tamanho do array "x". É criado um novo objeto JSON vazio "newJson".

Depois, é iniciado um loop "for" que percorre todas as chaves (horas) presentes no objeto JSON original, ordenando-as em ordem crescente. Dentro do loop, é criada a variável "key" que recebe a chave atual do loop. Em seguida, é adicionado ao objeto "newJson" um novo objeto com a chave atual "key" e o valor correspondente aos dados da estação de Bragança na hora indicada pela chave.

Por fim, é atualizado o valor da mensagem "msg.payload" com o novo objeto JSON "newJson", que contém apenas os valores filtrados da estação de Bragança. A mensagem é então retornada para o próximo nó do fluxo.

Depois de gerado o objeto JSON para os valores exclusivamente da estação de Bragança, cada um dos fluxos vai ter uma nova função para recolher os valores de temperatura e humidade, gerando as respetivas mensagens.

- Função de mensagem para a temperatura atual:

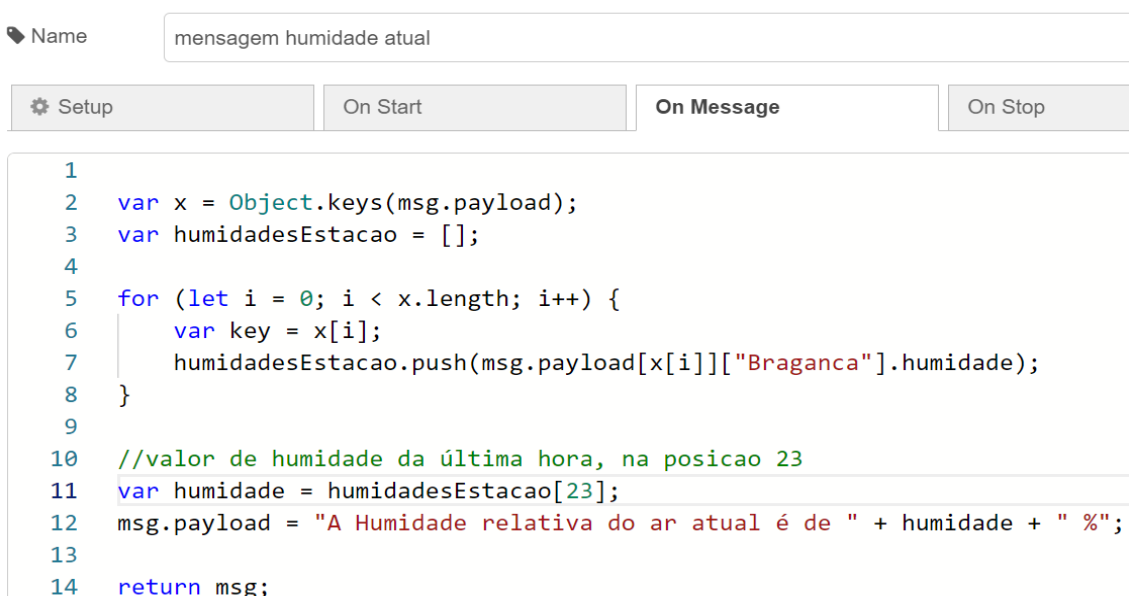


The screenshot shows a Node-RED function node configuration. The 'Name' field is set to 'mensagem temperatura Atual'. The 'On Message' tab is selected. The code is as follows:

```
1
2  var x = Object.keys(msg.payload);
3  var temperaturasEstacao = [];
4
5  for (let i = 0; i < x.length; i++) {
6    var key = x[i];
7    temperaturasEstacao.push(msg.payload[x[i]]["Braganca"].temperatura);
8  }
9
10 //valor de temperatura da última hora, na posicao 23
11 var temperatura = temperaturasEstacao[23];
12 msg.payload="A temperatura atual é de " + temperatura + " graus celsius";
13
14 return msg;
```

Figura 26: Nó de função para mensagem de temperatura atual.

- Função de mensagem para a Humidade atual:



The screenshot shows a Node-RED function node configuration. The 'Name' field is set to 'mensagem humidade atual'. The 'On Message' tab is selected. The code is as follows:

```
1
2  var x = Object.keys(msg.payload);
3  var humidadesEstacao = [];
4
5  for (let i = 0; i < x.length; i++) {
6    var key = x[i];
7    humidadesEstacao.push(msg.payload[x[i]]["Braganca"].humidade);
8  }
9
10 //valor de humidade da última hora, na posicao 23
11 var humidade = humidadesEstacao[23];
12 msg.payload = "A Humidade relativa do ar atual é de " + humidade + " %";
13
14 return msg;
```

Figura 27: Nó de função para mensagem de humidade atual.

Neste nó de função, estamos a receber como entrada um objeto JSON que foi previamente filtrado e contém os dados de temperatura da estação de Bragança para as últimas 24 horas.

Primeiro, usamos o método `Object.keys()` para obter as chaves do objeto JSON, que correspondem às horas em que os dados foram registados. Em seguida, percorremos todas as chaves usando um loop `for` e para cada chave, adicionamos o valor de temperatura e humidade correspondente para a estação de Bragança num array chamado `temperaturasEstacao` e `humidadesEstacao`, respetivamente para cada uma.

Por fim, acedemos aos valores de temperatura e humidade da última hora, que está armazenado na posição 23 de cada um dos arrays (`temperaturasEstacao` e `humidadesEstacao`). Usando este valor, construímos uma mensagem de resposta que informa ao utilizador a temperatura e humidade atuais em Bragança, pelo nó de debug, e é enviada para o nó de saída de áudio. É neste nó que a mensagem será reproduzida em voz sintetizada para o utilizador.

Abaixo, nas figuras, estão apresentados os resultados obtidos pelo fluxo Node-RED para a temperatura e humidade atuais.

Nesta primeira figura 28, a temperatura atual corresponde ao valor registado na última hora disponível nos dados fornecidos pela API, igual a 16.8 graus celsius.

```
▶ 2023-05-04T13:00: object
▶ 2023-05-04T14:00: object
▼ 2023-05-04T15:00: object
  ▼ Braganca: object
    intensidadeVentoKM: 13.7
    temperatura: 16.8
    radiacao: 1177.3
    idDireccVento: 7
    precAcumulada: 0
    intensidadeVento: 3.8
    humidade: 53
    pressao: 1019.9

04/05/2023, 17:22:05 node: debug 18
temperatura atual : msg.payload : string[43]
"A temperatura atual é de 16.8 graus celsius"
```

Figura 28: Nó de função para mensagem de temperatura atual.

Para os valores de humidade, o último valor é igual a 53% como corresponde a seguinte figura 29.

```
▶ 2023-05-04T14:00: object
▼ 2023-05-04T15:00: object
  ▼ Braganca: object
    intensidadeVentoKM: 13.7
    temperatura: 16.8
    radiacao: 1177.3
    idDireccVento: 7
    precAcumulada: 0
    intensidadeVento: 3.8
    humidade: 53
    pressao: 1019.9
```

---

```
04/05/2023, 17:20:59 node: debug 15
humidade atual : msg.payload : string[41]
"A Humidade relativa do ar atual é de 53 %"
```

Figura 29: Nó de função para mensagem de humidade atual.

Por fim, conclui-se que com as devidas operações, ações e parâmetros utilizados para realizar esta atividade, houve êxito nos desafios propostos, de tal forma que permitiu ganho significativo de aprendizado.

## Referências

- [Amazon AWS ]    AMAZON AWS: *What's API?* Disponível em: <https://aws.amazon.com/pt/what-is/api/>. Acesso em: 01 Mai. 2023.
- [ReadHat ]    READHAT: *What is a REST API?* Disponível em: <https://www.redhat.com/en/topics/api/what-is-a-rest-api/>. Acesso em: 01 Mai. 2023.
- [TOTVS 2020]    TOTVS: *Arquitetura REST: Saiba o que é e seus diferenciais*. Disponível em: <https://www.totvs.com/blog/developers/rest/#:~:text=0%20que%20%C3%A9%20REST%3F,projeto%20com%20interfaces%20bem%20definidas>. Acesso em: 01 Mai. 2023. 2020
- [W3 ]    W3: *WEB Services Architecture*. Disponível em: <https://www.w3.org/TR/ws-arch/>. Acesso em: 01 Mai. 2023.