**Ficha 4**

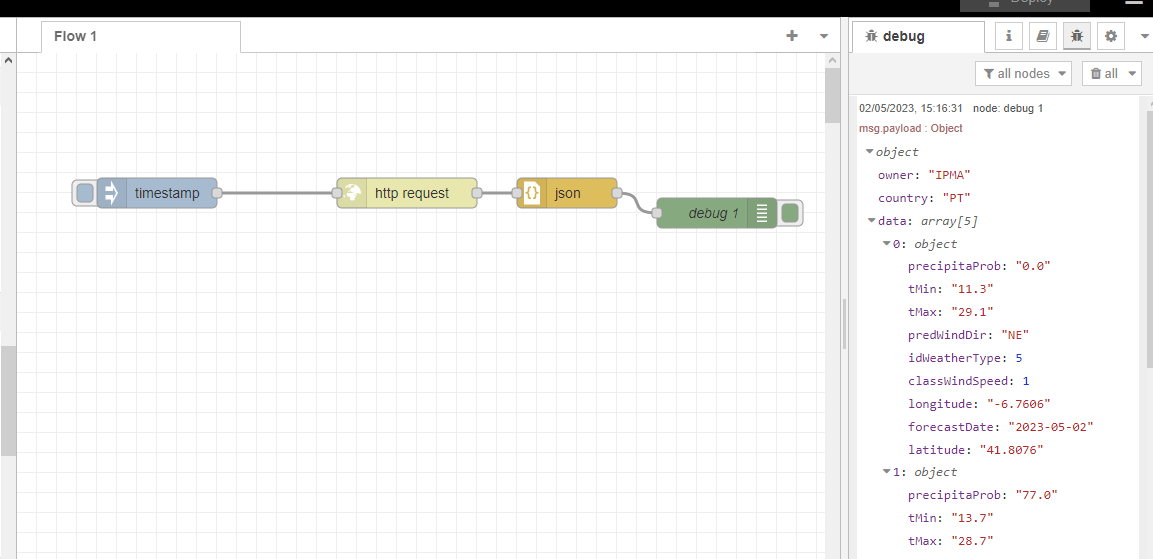
1. Serviços web são componentes de software que se comunicam com outros aplicativos pela internet seguindo padrões e protocolos definidos. Eles usam tecnologias como XML, HTTP e SOAP para permitir a comunicação e a interoperabilidade entre diferentes plataformas e linguagens de programação. O objetivo é fornecer uma interface padronizada e acessível para que diferentes aplicativos possam interagir e trocar informações de forma confiável, segura e escalável. Essa abordagem é amplamente usada na integração de sistemas empresariais, comércio eletrônico, redes sociais e IoT. O documento da arquitetura de serviços web do W3C define as melhores práticas e padrões para a criação e implementação de serviços web interoperáveis e escaláveis.

**B)** Interfaces de Programação de Aplicações, ou APIs (do acrônimo em inglês "Application Programming Interfaces"), são conjuntos de protocolos, rotinas, padrões e ferramentas que permitem que diferentes aplicativos se comuniquem e compartilhem dados e funcionalidades uns com os outros. Uma API atua como um intermediário entre diferentes componentes de software, fornecendo uma maneira padronizada e segura para que eles interajam e troquem informações. As APIs podem ser usadas para integrar serviços e aplicativos de terceiros em um aplicativo existente, permitir que diferentes aplicativos compartilhem recursos e funcionalidades, criar extensões personalizadas para aplicativos ou permitir que diferentes aplicativos se comuniquem e compartilhem dados em tempo real.

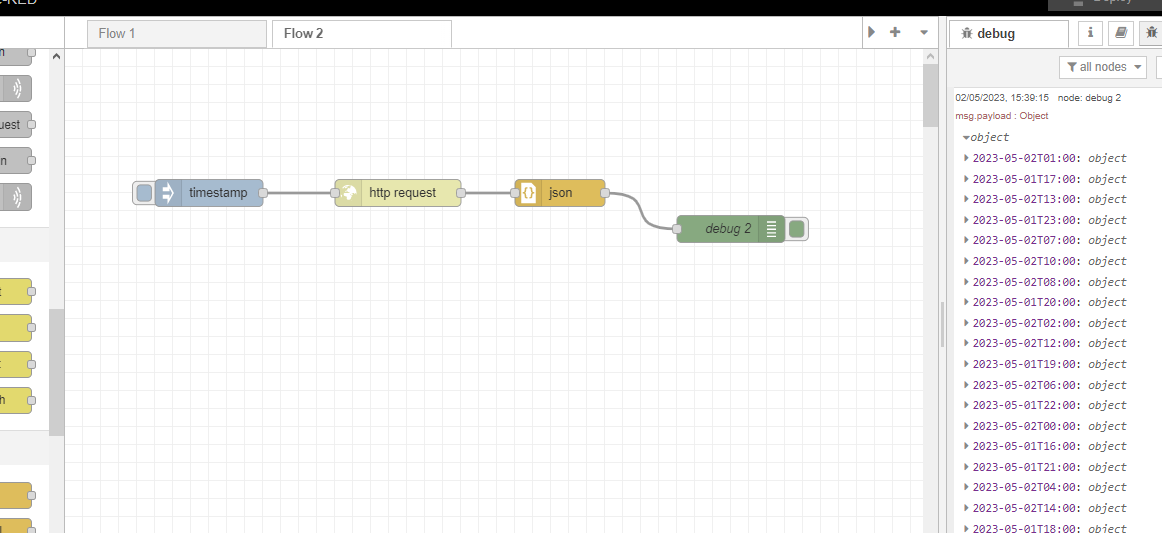
1. Uma API RESTful é uma interface de programação de aplicações que segue os princípios da arquitetura REST. Ela permite que os desenvolvedores usem métodos HTTP padrão, como GET, POST, PUT e DELETE, para realizar operações em recursos identificados por URI exclusivos. As respostas são retornadas em um formato comum, geralmente JSON ou XML. Essa abordagem é flexível, escalável e amplamente usada na construção de aplicativos web, dispositivos móveis e sistemas de IoT.

**2)**

B) ele retorna dados em json mostra a probabilidade de chuva, local e etc, nos próximas 5 dias.



1. Retorna dados de todas estações meteorológicas de Portugal das últimas 24h.



1. Desenvolva uma interface com o Node-RED que apresente num gráfico a temperatura e humidade disponível na estação meteorológica de Bragança nas ultimas 24 horas. Também adicione na interface a probabilidade de chuva, temperatura máxima e temperatura mínima para os próximos 5 dias. Obs: A disposição e organização da interface fica a critério do grupo.

R: fazer o request dentro da API e realizando o filtro utilizando um nó function e exibindo em gráfico chat no node-red.

let data = msg.payload;

let keys = Object.keys(data);

let sortedKeys = keys.sort((v1, v2) => new Date(v1).getTime() - new

    Date(v2).getTime());

let newData = {};

let stationCode = '1200575';

newData.labels = sortedKeys;

newData.series = ['Temperature', 'Humidity'];

newData.data = [];

let tempValues = [], humValues = [];

for (let k of sortedKeys) {

    tempValues.push(data[k][stationCode]['temperatura']);

    humValues.push(data[k][stationCode]['humidade']);

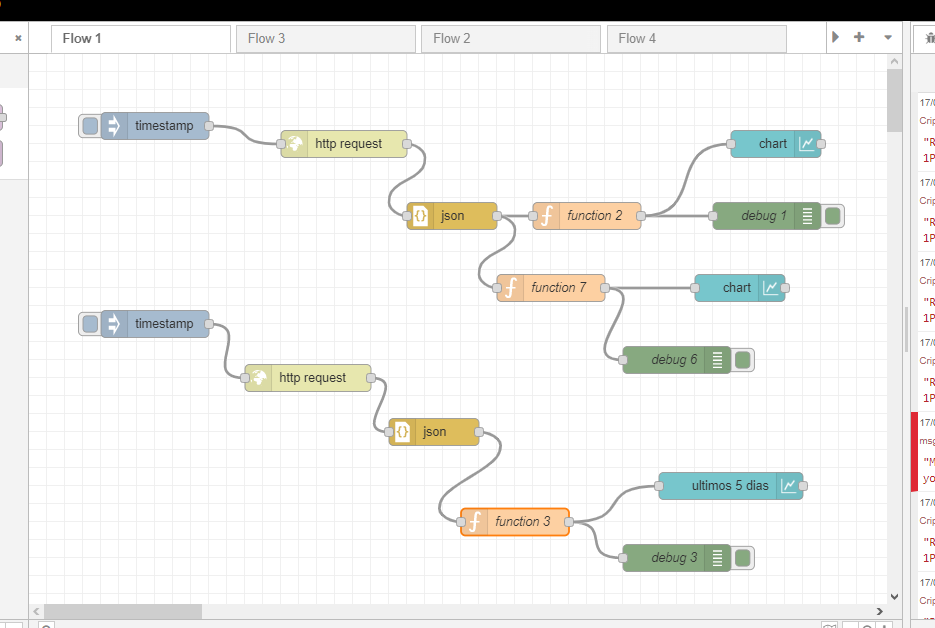
}

newData.data.push(tempValues);

newData.data.push(humValues);

msg.payload = [newData];

return msg;



Codigo javascript para filtrar dados de probabilidade de chuava, tem-max e temp-mini dos próximos dias da API.

let chartData = {};

let timestampStr = msg.payload['dataUpdate'];

let date = new Date(timestampStr);

let dates = [];

for (let i = 0; i < 5; i++) {

    dates.push(new Date(date.setDate(date.getDate() +

        1)).toISOString().split('T')[0]);

}

chartData.labels = dates;

chartData.series = ['Probabilidade de chuva', 'Temperatura máxima',

    'Temperatura mínima'];

chartData.data = [];

let rainProb = [], tempMax = [], tempMin = [];

for (let data of msg.payload.data) {

    rainProb.push(data.precipitaProb);

    tempMax.push(data.tMax);

    tempMin.push(data.tMin);

}

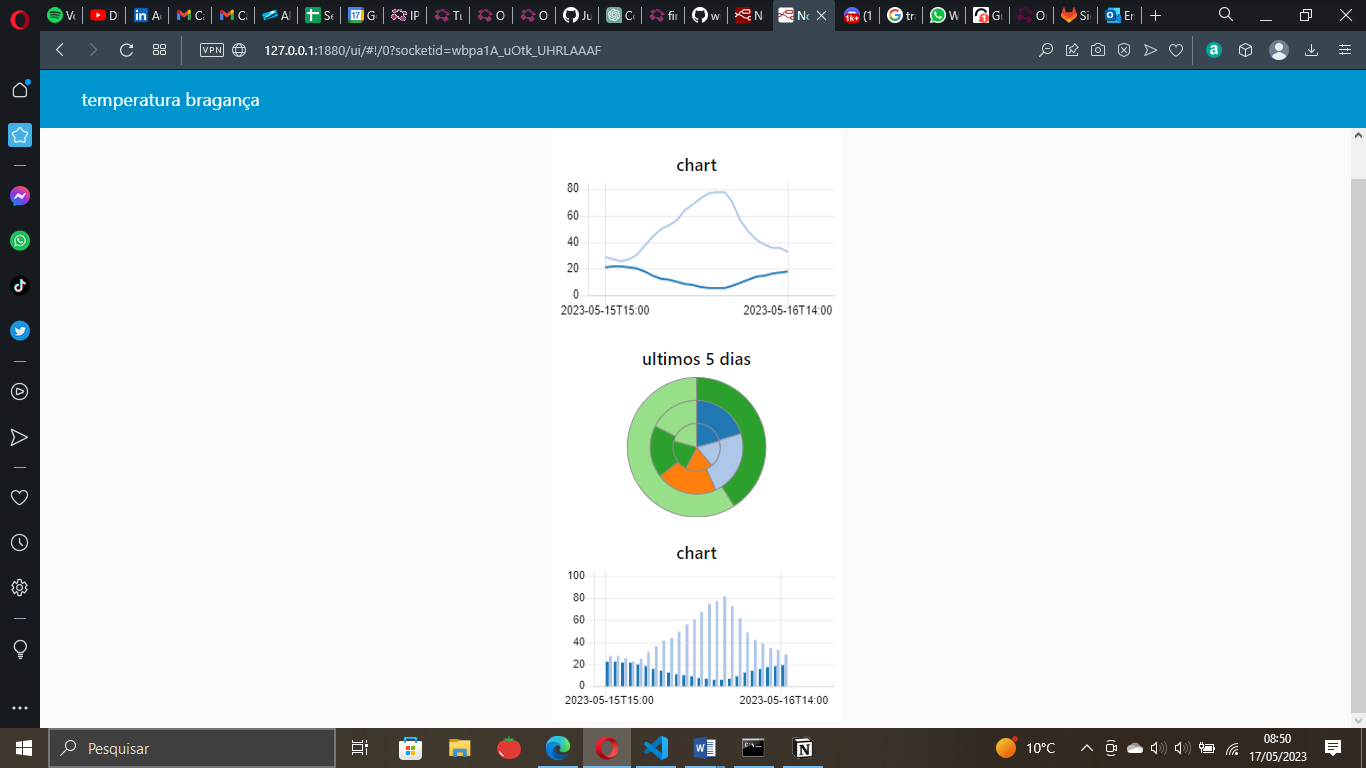
chartData.data.push(rainProb);

chartData.data.push(tempMax);

chartData.data.push(tempMin);

msg.payload = [chartData];

return msg;



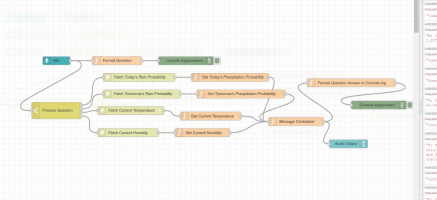
**3)**

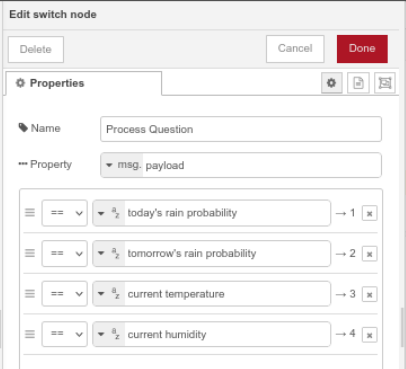
**(c) (\*)** Conforme os conhecimentos adquiridos nos tópicos acima, desenvolva o assistente meteorológico comandado por voz, que deve responder as seguintes questões: 1. Input usuário: “Previsão de Chuva Hoje” – Resposta Assistente: "Probabilidade de Chuva hoje é de X %". O valor de X é obtido da API IPMA utilizando o serviço "Previsão Meteorológica Diária até 5 dias agregada por Local". 2. Input usuário: “Previsão de Chuva Amanhã” – Resposta Assistente: "Probabilidade de chuva para amanhã é de Y % ". O valor de Y é obtido da API IPMA utilizando o serviço "Previsão Meteorológica Diária até 5 dias agregada por Local". 3. Input usuário: “Temperatura Atual” – Resposta Assistente: "A temperatura atual é de Z graus Celsius". O valor de Z é obtido da API IPMA utilizando o serviço "Observação Meteorológica de Estações (dados horários, últimas 24 horas)". 4. Input usuário: “Humidade Atual” – Resposta Assistente: "A humidade relativa do ar atual é de J %". O valor de J é obtido da API IPMA utilizando o serviço "Observação Meteorológica de Estações (dados horários, últimas 24 horas)".

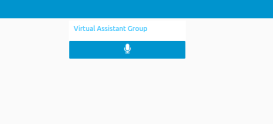
**R:** utilizei os componentes de microfone e saída de áudio no Node-RED. Também integrei uma API para realizar a conversão de comandos de voz em texto.

Durante a atividade, capturava o áudio por meio do microfone e enviava para a API, que retornava o texto correspondente à fala. Em seguida, utilizei uma estrutura de controle chamada "switch" para comparar as frases obtidas com diferentes opções pré-definidas.

Essa abordagem permitiu criar diferentes caminhos no fluxo do Node-RED, de acordo com as frases reconhecidas. Cada comando de voz correspondente a um caso no "switch" acionava a execução de ações específicas ou a resposta personalizada ao usuário.

****

****

****