**MANUAL DE UTILIZAÇÃO DHT11 / Node.js - Passo a Passo**

**Grupo Plot Twist**

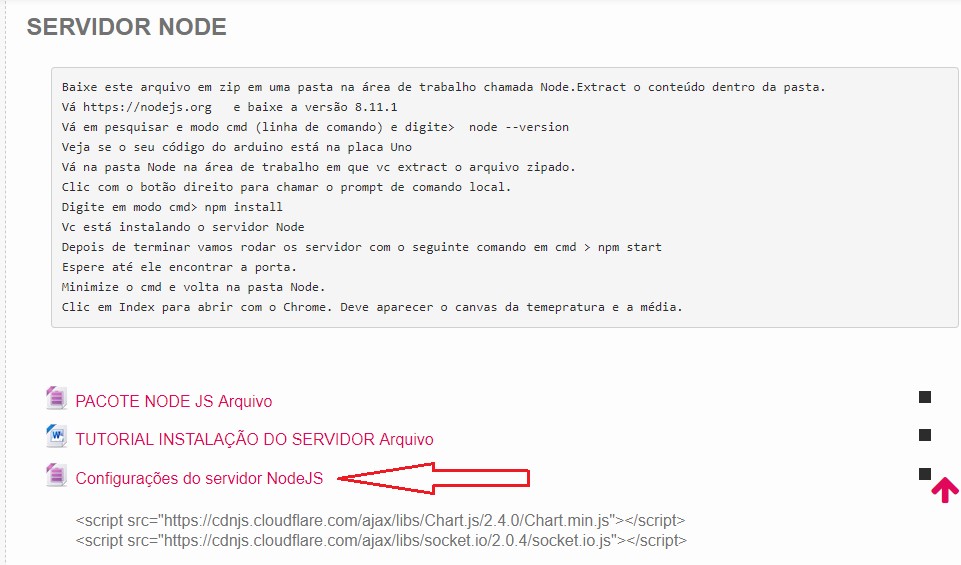
|  |  |
| --- | --- |
| **Nomes:** | **RA:** |
| Eduardo Kauan | 01191042 |
| Felippe Muniz | 01191038 |
| Marcelo Antony | 01191061 |
| Paulo Costa | 01191128 |
| Vitória Berniz | 01191126 |

PASSO 1:

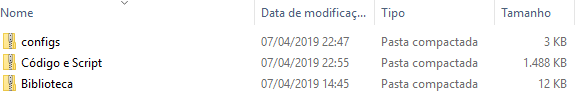
BAIXAR O CONTEÚDO A SER UTILIZADO

No Moodle, baixe o conteúdo do nosso grupo os arquivos Biblioteca.zip e Código e Script.zip que estão dentro do Tópico do Manual de Utilização;

Volte no Moodle e agora baixe as *Configurações do Servidor Node JS*, como na imagem:



Para facilitar, reúna todo o conteúdo necessário baixado em um só local:

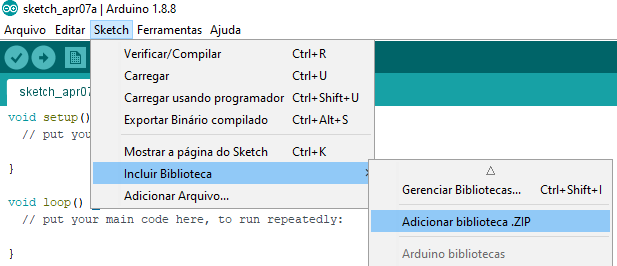


PASSO 2:

CONFIGURAR BIBLIOTECA DO ARDUINO

- Abra o programa do Arduino;

**Vá em:** Sketch **>** Incluir Biblioteca **>** Adicionar Biblioteca .ZIP

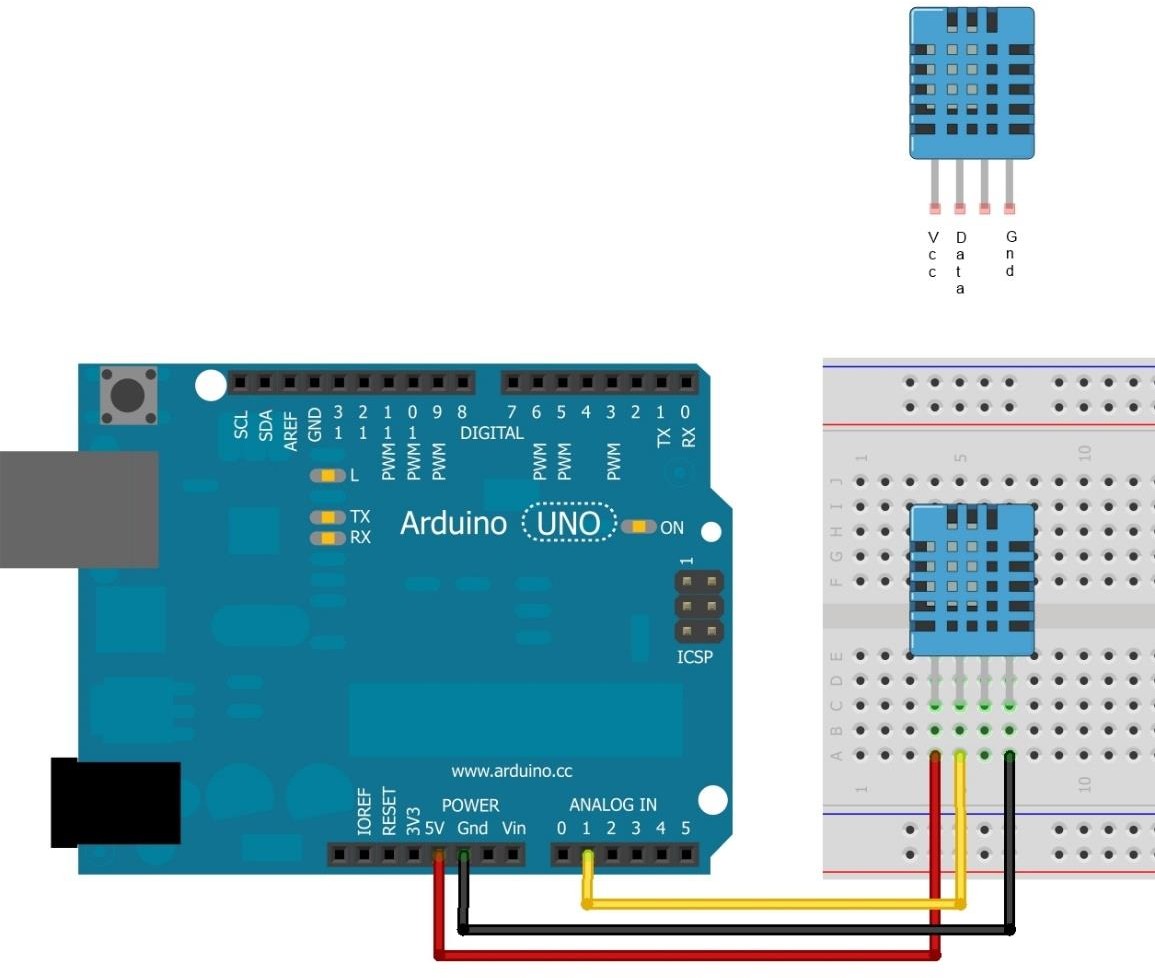


Selecione nosso arquivo Biblioteca.zip que foi baixado ►

*Ao clicar em Abrir, a Biblioteca estará instalada e configurada.*

PASSO 3:

CONEXÃO E MONTAGEM DO SENSOR NO ARDUINO



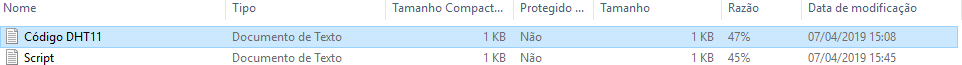
Realize a montagem dos fios na placa de acordo com a imagem e instrução:

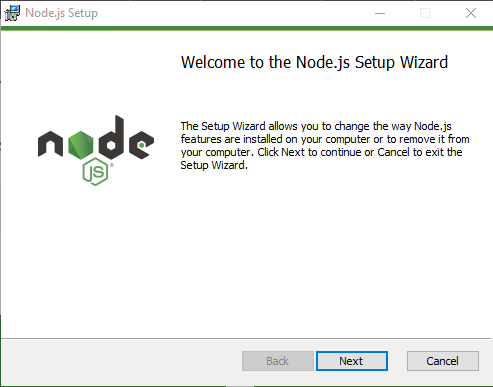
* Conecte o cabo do 1º pino à voltagem 5V;
* Conecte o cabo do 2º pino à porta que será utilizada, em nosso caso ***Analógica A1***.
* O 3º pino deve estar vazio;
* Conecte o cabo do 4º pino ao GND do Arduino.
* **Certifique-se se está tudo correto e conecte à placa USB do notebook.**
* Selecione e Verifique a porta **COM:** Na IDE do Arduino, em Ferramentas **>** Porta

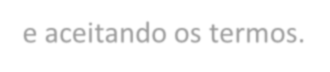
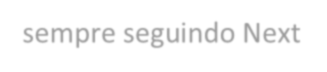
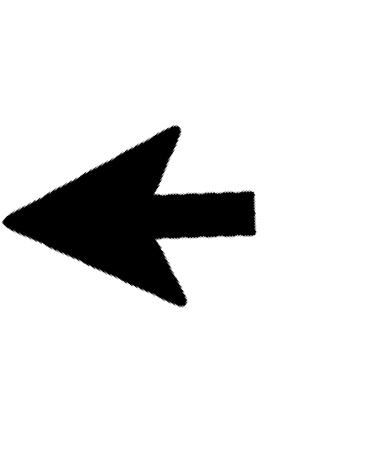
PASSO 4:

RODANDO O CÓDIGO NO ARDUINO

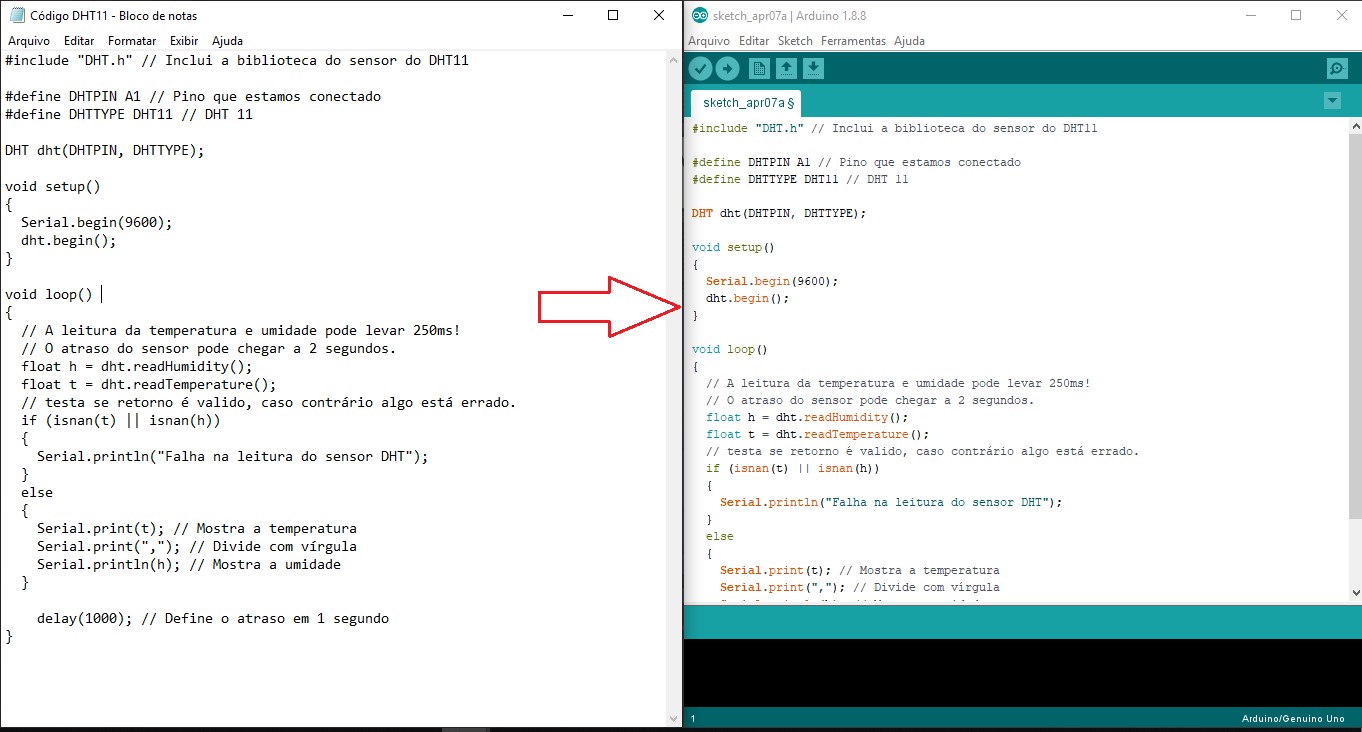
Cole o código do Arduino que está presente na pasta Código e Script.zip:



Aperte *CTRL+A* no Arduino para Selecionar tudo, e cole o código do Bloco de Notas (Código do Sensor DHT11.txt)



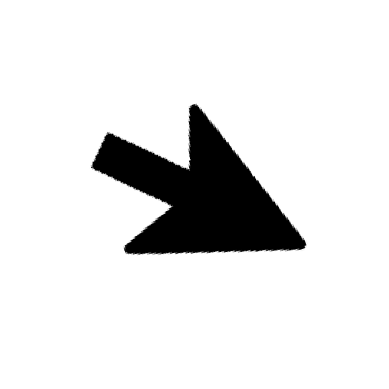
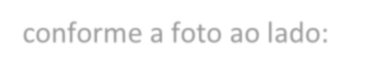
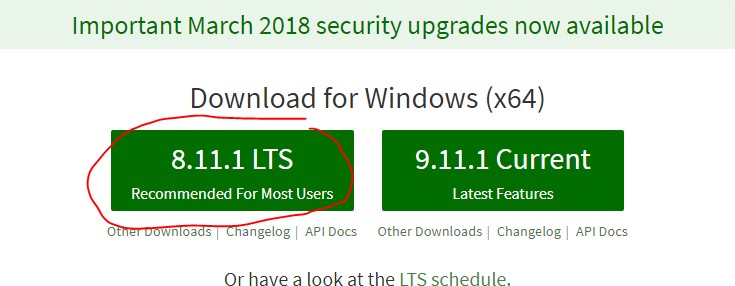
Instala o aplicativo, sempre seguindo Next e aceitando os termos.



- Feito isso, Verifique e Compile o Código.

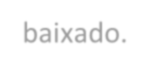
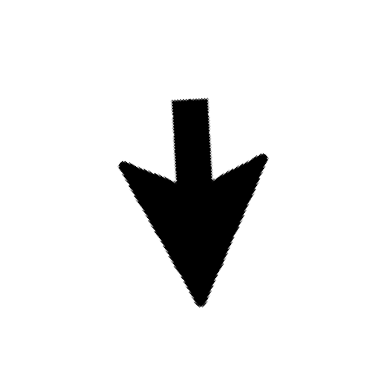
PASSO 5:

BAIXAR E INSTALAR O NODE JS (Caso não esteja)

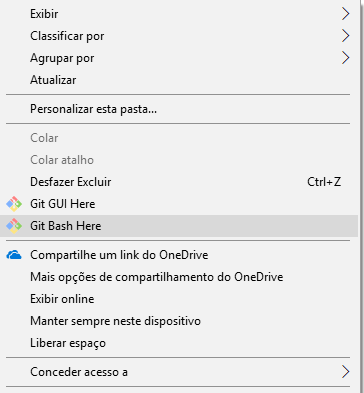


Fazer o download

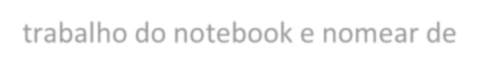
conforme a foto ao lado:



Aguardar o arquivo ser baixado.

***Para certificar de que o Node já está instalado, abra o CMD e digite node –version Dê enter. Caso esteja aparecerá o número da versão, caso contrário, terá de instalar.***

PASSO 6: CONFIGURAÇÃO DO SERVIDOR NODE JS

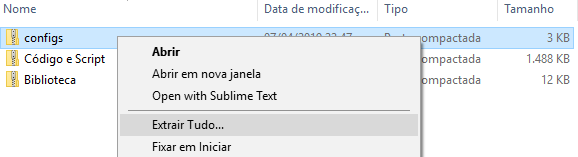


Criar uma diretório na área de

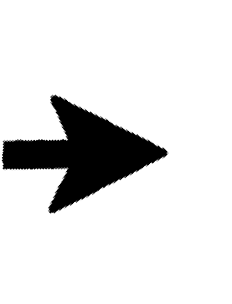
trabalho do notebook e nomear de “Node.Extract”

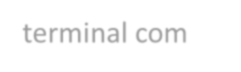
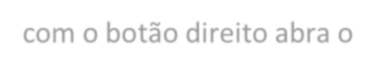


Agora, extraia o conteúdo da pasta configs.zip para o conteúdo da pasta *Node.Extract* que foi criada.



Depois:





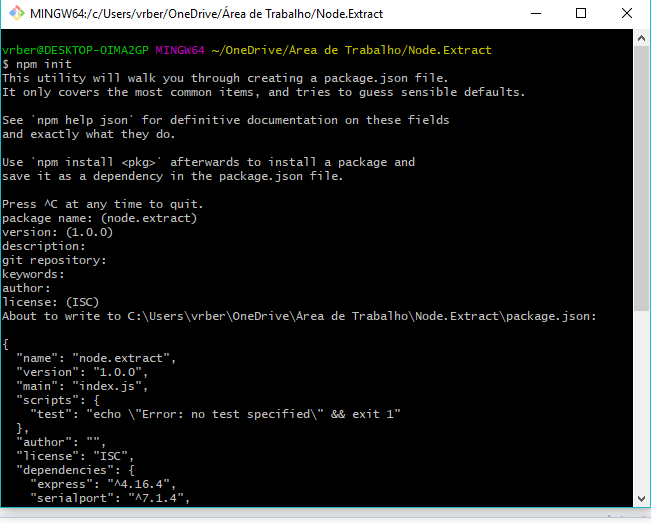
Abra o diretório e dentro dele

com o botão direito abra o terminal com

“Git Bash Here”

Com o terminal do Git aberto, digite o comando:

## npm init



Agora digite o comando no terminal:

## npm install express socket.io serialport --save

***Dê enter.***

Nesta tela dê enter em todas as etapas até aparecer:

Is this OK? (yes)

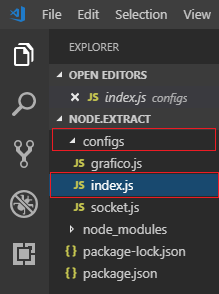
Dê enter novamente.

Agora, certifique-se de que o arquivo *package.json* foi criado dentro da pasta Node.Extract.

# Certifique-se que de foram criadas a pastas node\_modules e o arquivo package-lock.json na nossa pasta.

**Abra o Visual Studio Code.**

## Clique em File **>** New Folder **>** Selecione a pasta Node.Extract

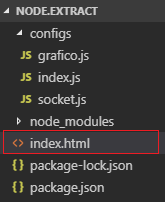
Abra o arquivo configs/index.js:

Procure a seguinte linha:



Edite COM7 para a porta COM correspondente que está sendo utilizada no Arduino.

*Obs: Vá salvando a cada edição com CTRL+S.*

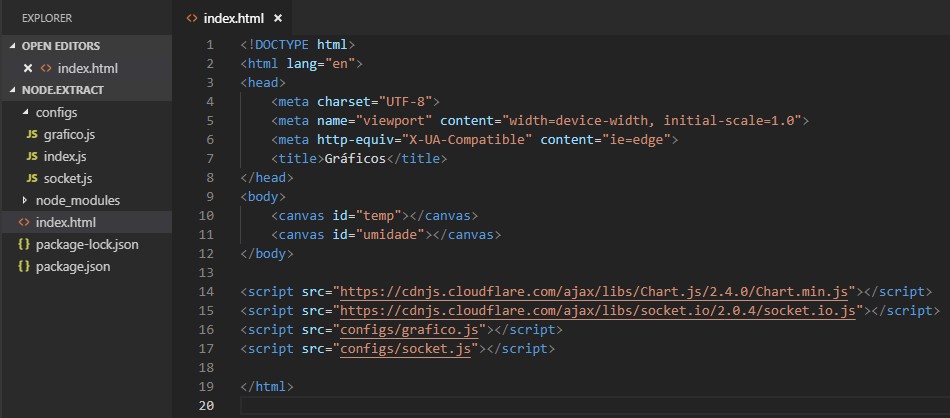
Agora, crie um novo arquivo chamado index.html para ficar dentro da nossa pasta Node.Extract:

Utilize a exclamação “!” para criar um código padrão HTML.

Agora, vá até o nosso conjunto de arquivos e abra o Código e Script.zip, depois abra o Script.txt

*Copie e cole os scripts seguindo o ‘mini tutorial’ em comentário da Scripts.txt*

Sua index.html deverá ficar assim:



PASSO 7:

MOSTRANDO OS VALORES NO GRÁFICO

Para mostrar os valores de **temperatura**, deve-se fazer uma alteração em nosso código do Arduino.

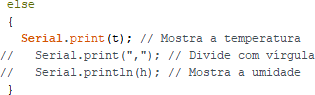
Comente o código no **programa do Arduino** correspondentes à umidade. Serial.print(t); // Mostra a temperatura

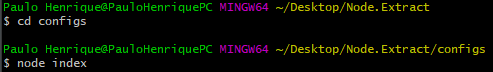
// Serial.print(","); // Divide com vírgula

// Serial.println(h); // Mostra a umidade

}

Na parte que estamos modificando, no Arduino terá que ficar assim:



Agora, retorne à janela terminal do Git Bash e digite:

cd configs

Logo em seguida, digite:

node index

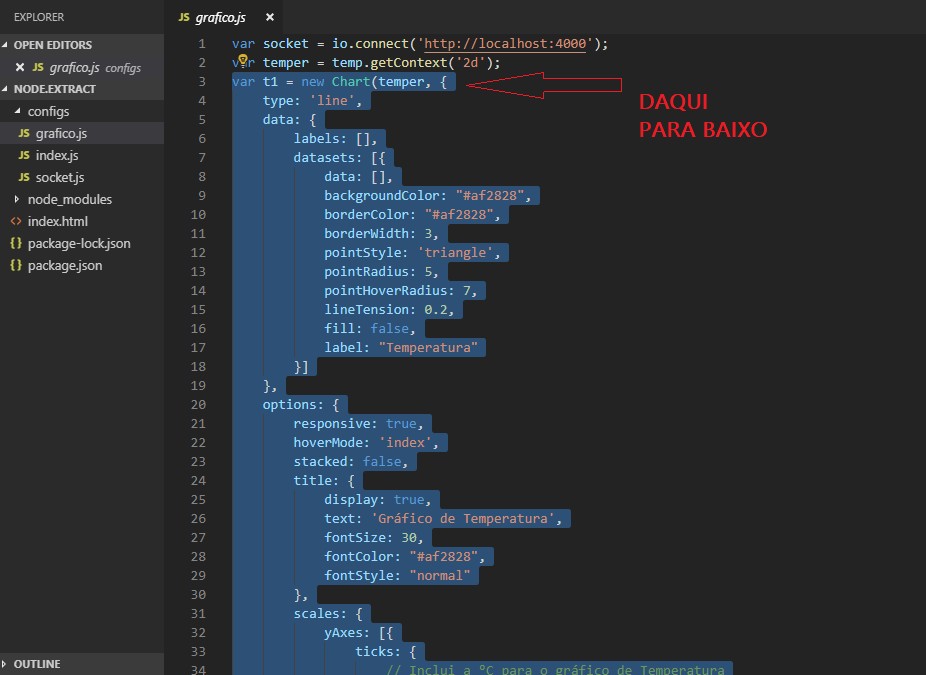
 ***Ao digitar node index, deverá mostrar as temperaturas no Terminal logo abaixo.***

Com isso, abra o index.html da *Node.Extract* que a temperatura deverá estar sendo mostrada corretamente.

**Gerando o Gráfico de Umidade**

 Para gerarmos o gráfico de umidade, você irá precisar alterar o gráficos.js no Visual Studio Code.

Copie todo o código do gráficos.js a partir da 3ª linha, e cole-o logo abaixo do mesmo código para podermos alterá-lo e transformá-lo em umidade:



Cole (CTRL+V) o código copiado para abaixo dessa mesma linha de código, dando um salto de uma linha. (l. 42 p/ 44)

Feito isso, só precisamos alterar os valores:

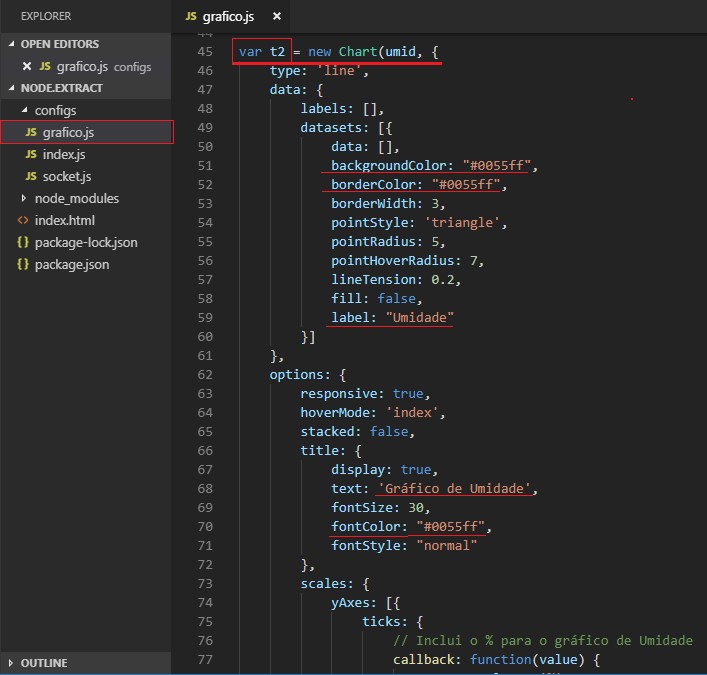
Altere tudo que está na esquerda p/ valor da direita do código colado abaixo no VSC

|  |  |
| --- | --- |
| **Temperatura** | **Umidade** |
| var t1 = new Chart( temper, { | var t2 = new Chart( umid, { |
| backgroundColor: “#af2828” | backgroundColor: “#0055ff” |
| borderColor: “#af2828” | borderColor: “#0055ff” |
| label: “Temperatura” | label: “Umidade” |
| text: “Gráfico de Temperatura” | text: “Gráfico de Umidade” |
| fontColor: “#af2828” | fontColor: “#0055ff” |
| return value + ‘ºC’ | return value + ‘%’ |

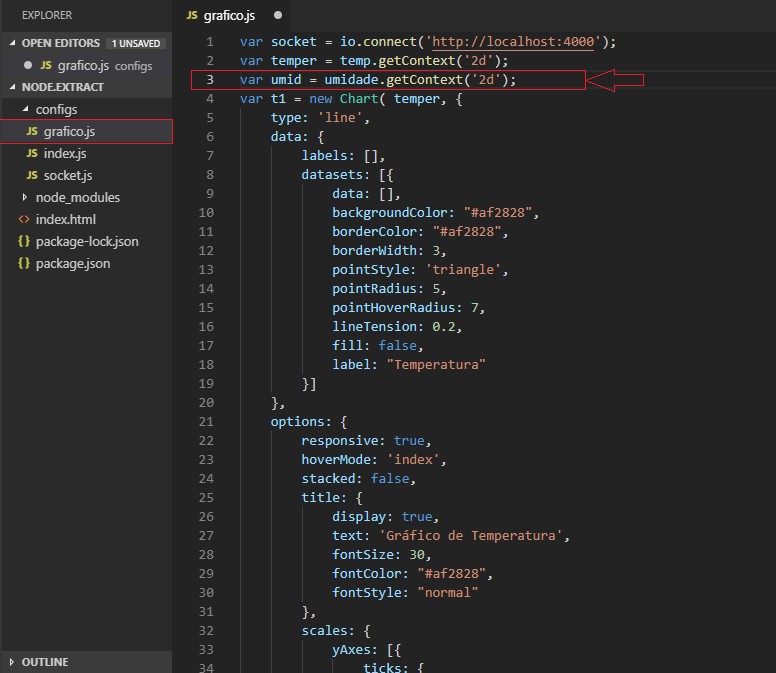
***Atenção: Corrigir as aspas se necessário caso for colar os valores da tabela.***

Deverá ficar assim:

(códigos correspondentes do gráfico um abaixo do outro)

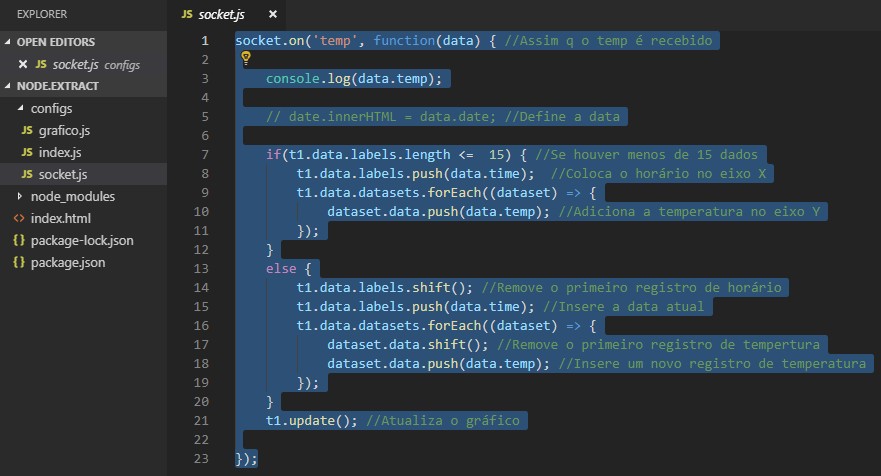


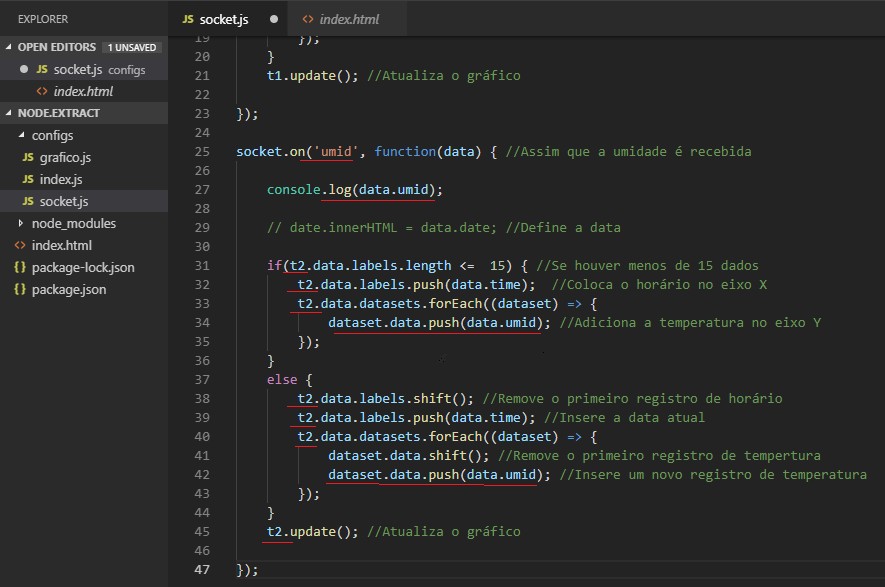
Feito isso, na parte de cima do gráfico.js, acrescente uma nova variável:

var umid = umidade.getContext('2d');

Ficando desse modo:

Agora, vá até o *socket.js* no lado esquerdo do Visual Studio Code e faça o mesmo procedimento de copiar e colar o código um abaixo do outro:



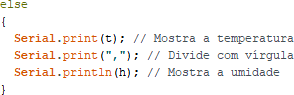
**Altere no código colado, todos os itens que possuem “temp” para “umid”, e todos os itens “t1” para “t2”.**

Deverá ficar assim:

 ***Certifique-se de cada uma das linhas contendo t1 e temp foram mudadas corretamente para t2 e temp, ou não funcionará.***

Feito isso, é necessário retornar o código do Arduino para seu estado original.

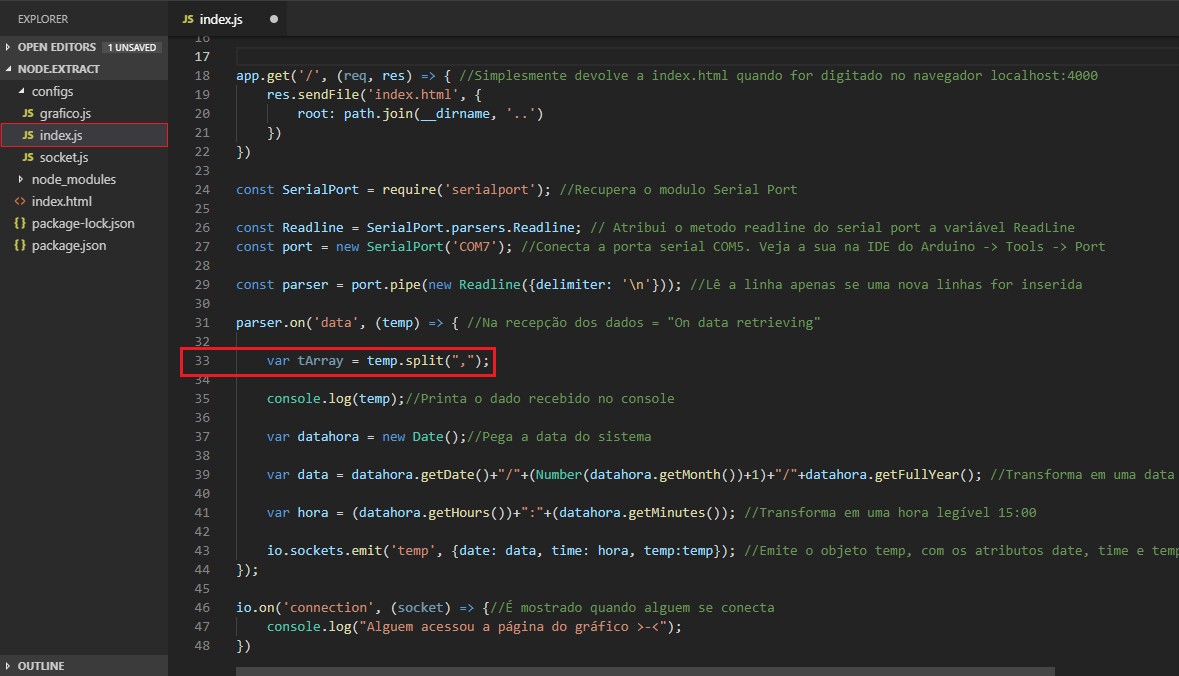
Retire os dois comentários feitos no Serial.print.

Ficando assim na parte modificada:

Para finalizar, volte no arquivo index.js onde iremos adicionar uma variável tA*rray* para tornar as duas temperaturas mostradas no Terminal Git sendo divididas por vírgula:

var tArray = temp.split(",");

Ficando desse modo:



Altere a *linha 43* io.sockets.emit(...) logo abaixo, para deixá-la como:

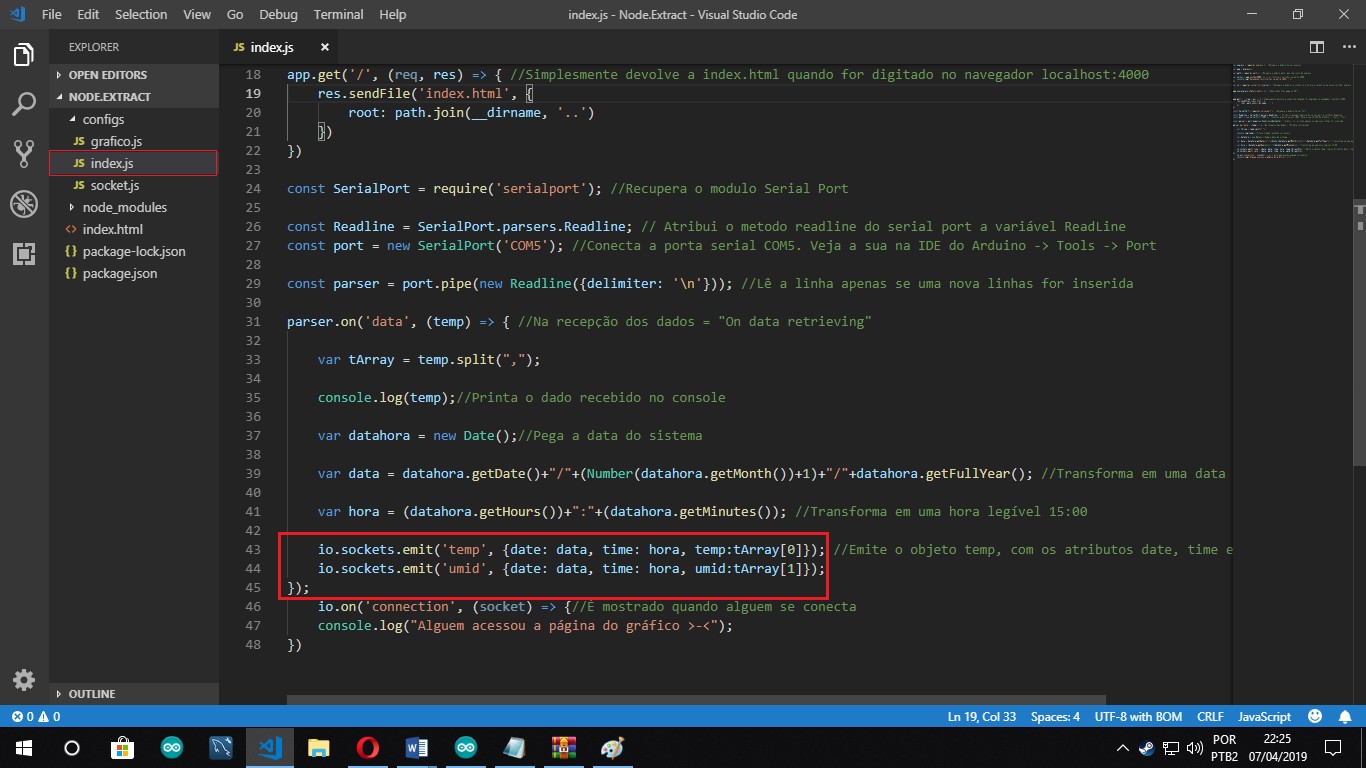
io.sockets.emit('temp', {date: data, time: hora, temp:tArray[0]});



***(Substituindo o segundo temp por tAarray e acrescentando o número [0])***

***Obs: o nome tArray é p/ evitar conflitos com qualquer array.***

Por fim, copie e cole o código do io.sockets.emit(...) acima para logo abaixo, alterando temp para umid, e acrescentando o [1] após o tArray, deixando o código desse modo:



***Certifique-se de que está igual como mostrado na imagem.***

Retorne à janela do Git Bash dentro da pasta configs, e execute novamente o comando:

node index

Agora, deverá estar aparecendo os valores de temperatura e umidade separados por vírgula, na própria janela do terminal.

 Abra o index.html da nossa pasta pelo navegador normalmente e os dois gráficos deverão estar sendo mostrados em conjunto, funcionando.

***Bandtec Digital School*** ©