

# **Guia Fantástico para Instalação do seu chatbot de whatsapp**

**Eric Raphael Reis**

# Introdução do Guia Fantástico

- Sejam todos e todas muito bem vindos ao guia fantástico para a construção do seu Chatbot para Whatsapp. Aqui você encontrará o passo a passo para deixar pronta a sua estrutura e atender aos seus clientes de forma 100% automática. As possibilidades são do tamanho da sua imaginação. Estou muito feliz por você estar aqui!
- O manual está dividido em etapas, eu pensei em cada uma delas enquanto um processo lógico onde a conclusão de cada módulo leva a um próximo necessariamente vinculado.
  - Primeiro vou te ensinar como baixar e instalar os programas necessários. Todo o processo de instalação será feito apenas uma única vez e depois seu trabalho será de apenas projetar e atualizar os seus funis de atendimento.
- Então você estará pronto para carregar um primeiro funil, o funil base, e logar seu whatsapp pelo qr-code para dar permissão ao robô realizar os atendimentos. Uma vez feita esta etapa, o robô estará 100% configurado e pronto.
- E agora com você especialista em todo o processo e já viu o robô funcionando, eu vou te explicar a lógica simples do bloco de código base que utilizei para você conseguir criar os seus próprios e arrear nas vendas.
- Também vou te conduzir na configuração da sua máquina virtual, a máquina virtual será usada para hospedar o nosso projeto por 24 horas ao dia na nuvem. Desta forma o robô continuará trabalhando

mesmo com o seu computador desligado ou celular fora de área/sem bateria. Trabalhar com a nuvem é excelente, pois nos possibilita liberdade, por outro lado precisamos estar atentos para evitar situações chatas de cobranças destas empresas pelos seus serviços de hospedagem. Fique tranquilo(a) pois eu fiz uma peneira dos principais serviços de nuvem gratuitos e te ensinarei como proceder. Você ainda poderá hospedar o seu robô em seu computador, mas ele ficará vulnerável a situações como queda de energia ou de conexão. A escolha fica contigo!

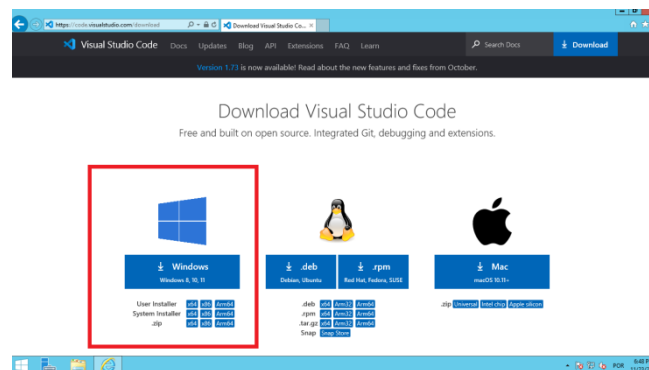
- Este guia será constantemente atualizado conforme novos projetos vão chegando. Bora começar!

# Modulo Start – Baixando e Instalando os programas necessários

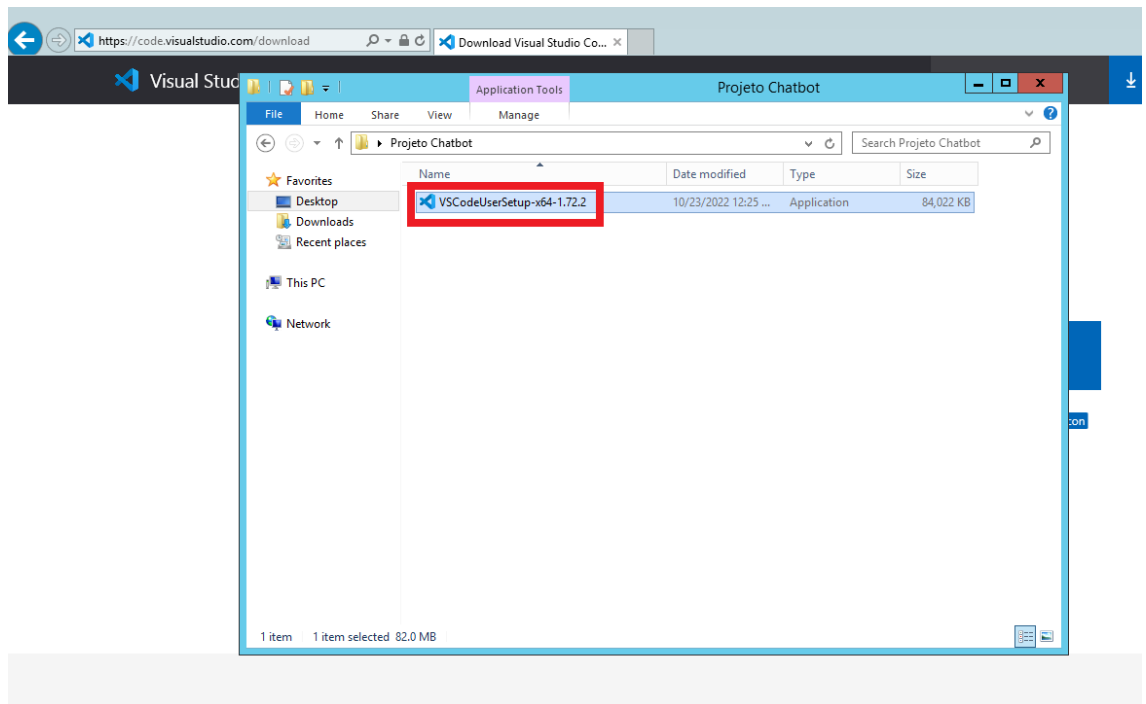
Todos os programas e pacotes que usaremos neste projeto são gratuitos e seus links disponibilizados junto com uma explicação.

- No exemplo deste guia, vou subir toda a estrutura numa máquina Virtual da Alibaba que ensinei no módulo “Como aderir e configurar uma máquina virtual e trabalhar com seu projeto na nuvem da Alibaba”. Você poderá seguir este passo a passo em qualquer computador com Windows, seja no seu próprio computador, seja numa máquina virtual hospedada.
- Vamos abrir o navegador para começarmos as festividades. No exemplo deste guia, usarei o Internet Explorer por ser o navegador padrão do Windows Server 2012, você pode trabalhar com o de sua preferência. **Dica: abra uma pasta fácil de encontrar para deixar todos os arquivos produzidos e baixados neste guia.**
- **É importante lembrar que o processo de instalação só precisa ser feito uma única vez.**
- Com o navegador aberto, acessamos o site do VS-Code descrito no link: <https://code.visualstudio.com/download>

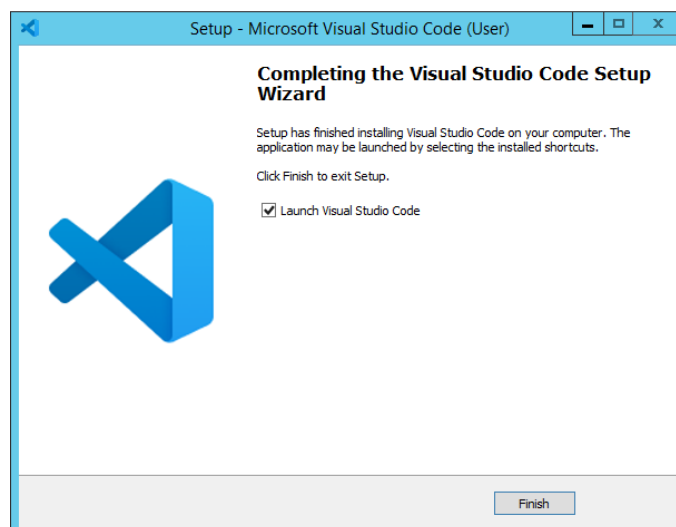
Ficará com esta tela:



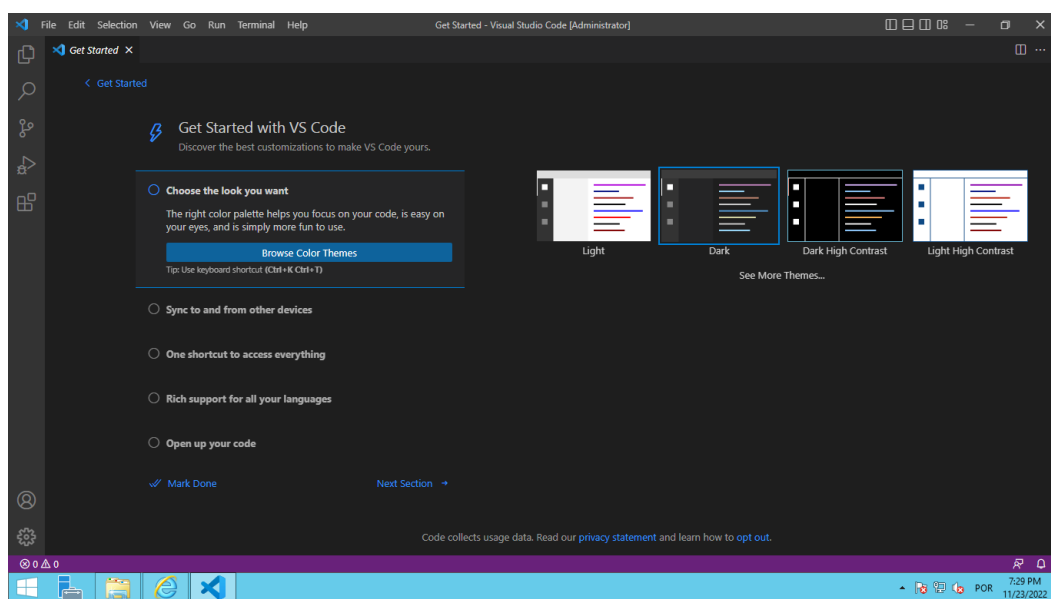
- Baixe o arquivo de instalação para Windows e deixe na sua pasta do projeto. Algumas máquinas virtuais não permitem que você baixe arquivos de outros websites, neste caso você vai copiar e colar os arquivos do seu computador para a máquina virtual.



- Confirme todos os passos de instalação do VS-Code da forma como são apresentados. Ao final do processo você verá uma tela de confirmação como essa:

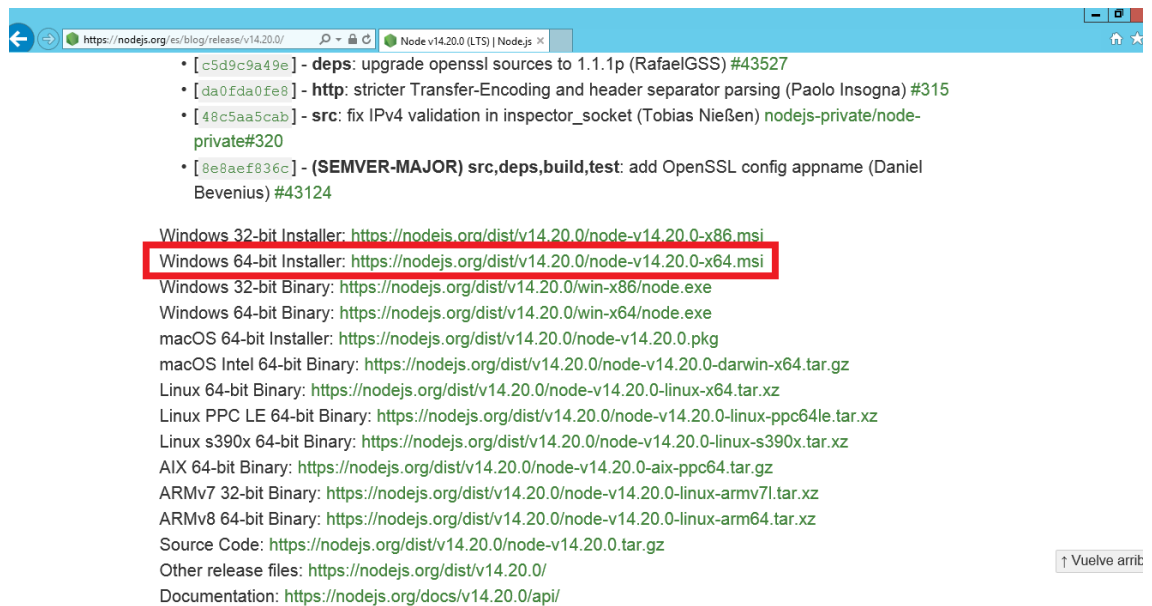


- Clique em “Finish” para terminar a instalação e o VS-Code.
- Breve descrição sobre o VS-Code: Este é um editor de código que vamos usar para abrigar nosso projeto. Pense no nosso projeto completo como uma casa montada e pronta para uma família. O VS-Code é o carrinho de mão que vai levar o cimento, areia e pedra em sua construção.
- A primeira tela, ao instalar o VS-Code se parecerá com essa tela:

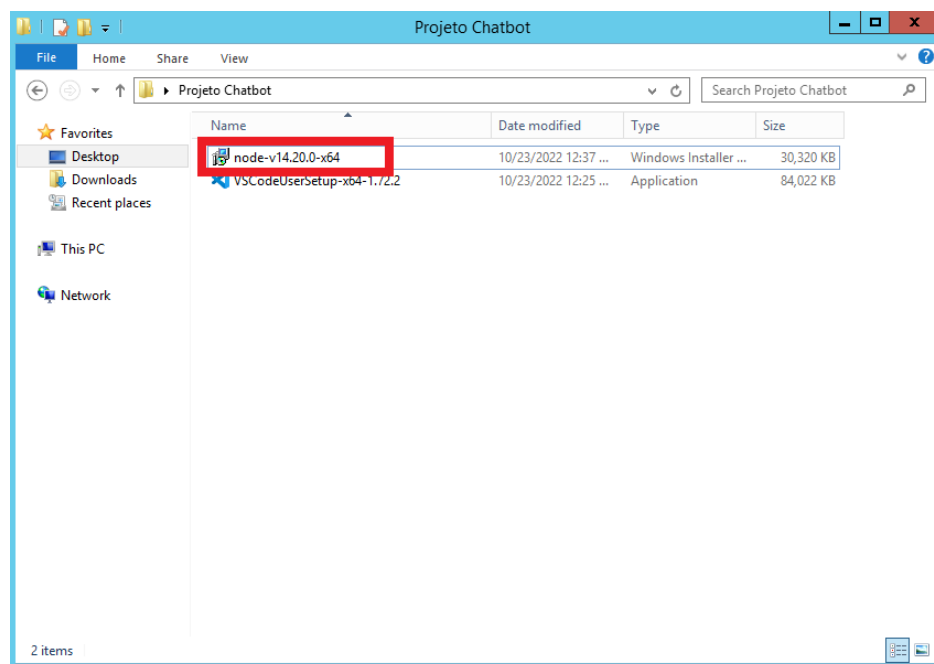


- Como neste exemplo estou a subir o projeto na nuvem do Alibaba, vou deixar sua linguagem em Inglês, você pode mudar para Portugues no seu computador local.
- Com o VS-Code instalado e aberto, vamos instalar e configurar o Node.js. Ele é o interpretador da linguagem que vamos utilizar, o Javascript, no exemplo da casa, o Node.js é o cimento, areia e pedra que são os materiais que vão solidificar nossa casa.

- Voltamos ao navegador para baixar o Node.js, colocamos o link no navegador: <https://nodejs.org/es/blog/release/v14.20.0/>

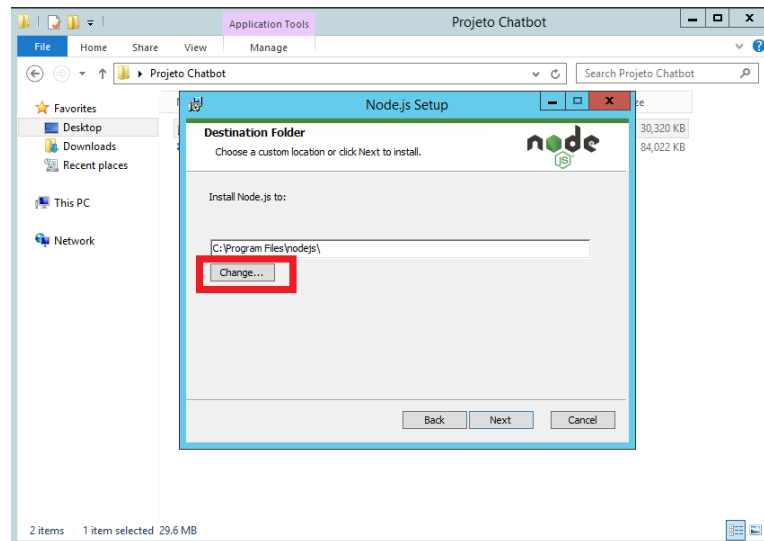


Rolamos para baixo um pouco até encontrarmos o link para o instalador de nosso Windows 64 bit (Todos os Windows Server e de 8 pra cima). Baixamos o arquivo de instalação e o movemos para a pasta do projeto.

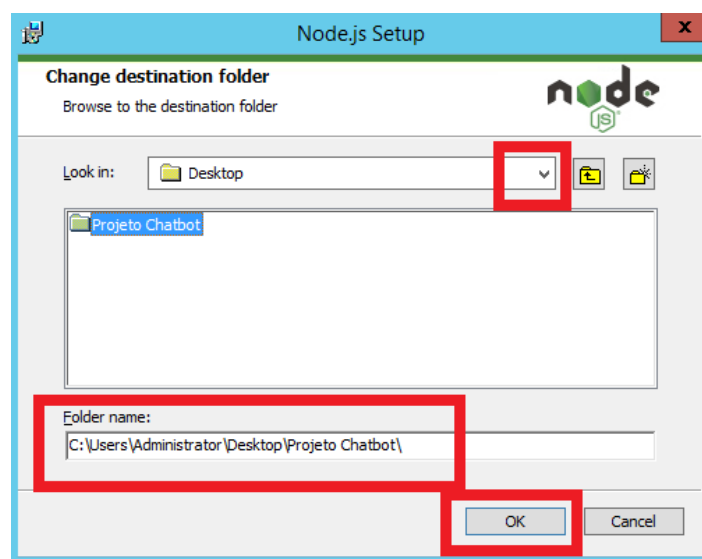


- Aceite e avance os passos do node.js, até a janela que te pedirá para escolher a pasta que abrigará a instalação do Node.js. Nela nós vamos mudar o local padrão para a nossa pasta do projeto. É possível instalar em outro lugar, mas para facilitar a navegação quando estivermos a subir a biblioteca neste tutorial, vamos usar a nossa pasta do projeto.

Para mudar clicamos em “Change” ou “Mudar”

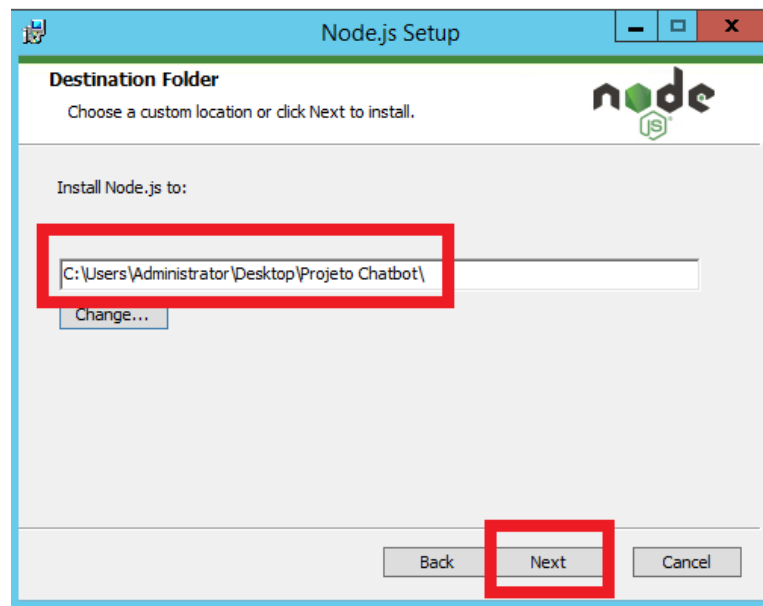


Escolhemos a pasta do projeto usando os navegadores como destacado na figura e clicamos em “OK”

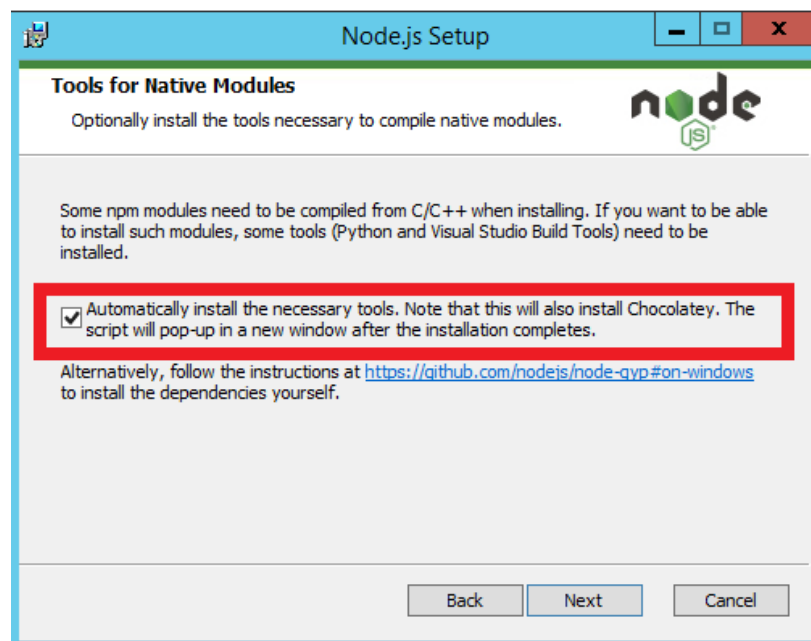




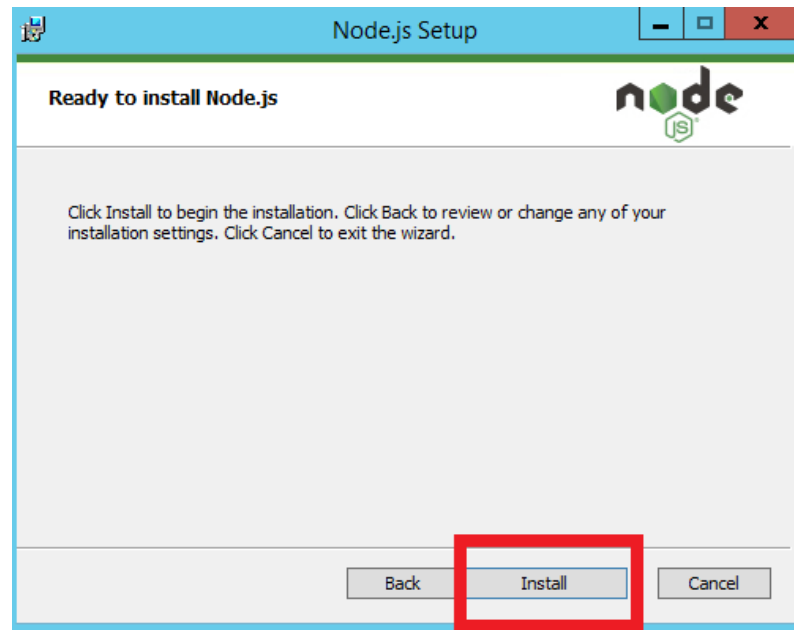
Com a mudança de pasta feita, nós clicamos em “Next” ou “Proximo” como na figura



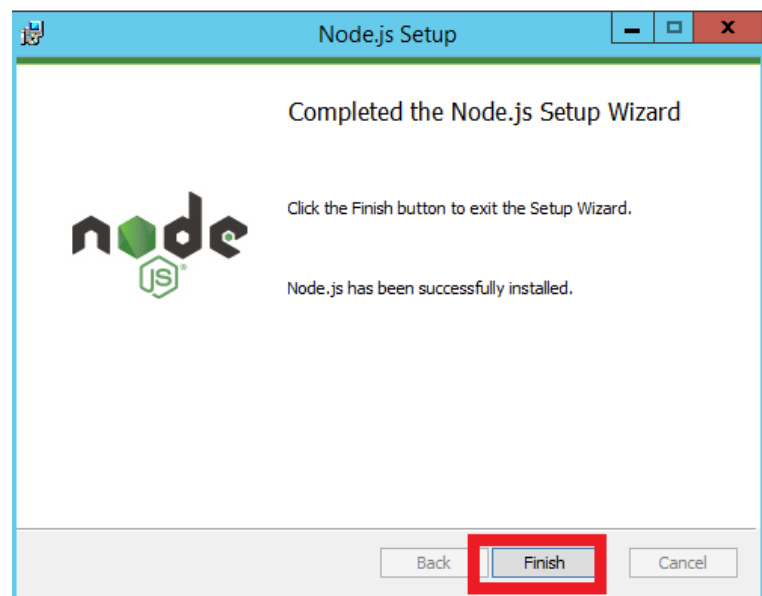
Enfim nós terminamos por deixar marcada a opção que permitirá ao instalador adicionar os demais pacotes pertinentes como destacado na figura abaixo



Por fim, clique em “Install” para dar inicio a instalação:



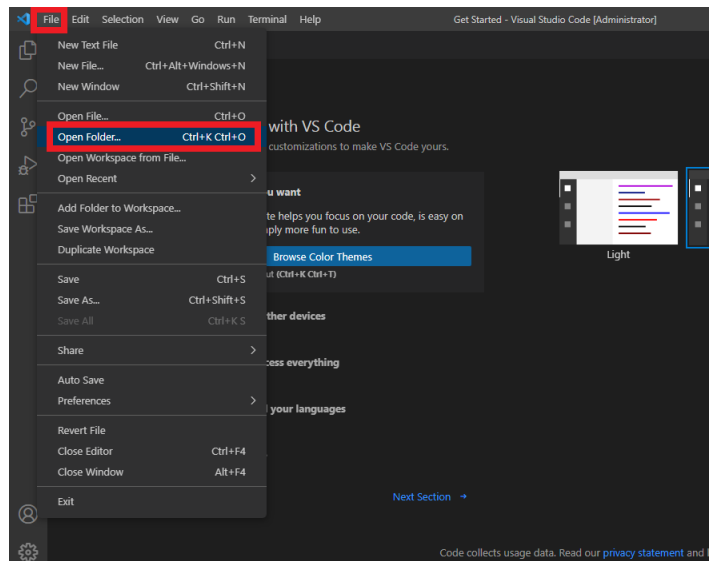
Ao finalizar você verá uma tela assim, basta clicar em “Finish”:



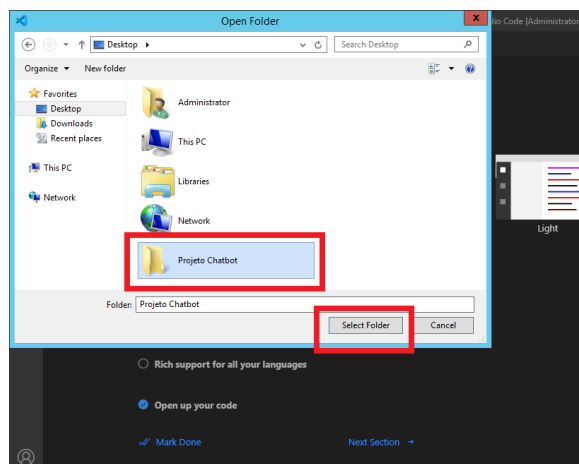
- Ele pode abrir um prompt para terminar de instalar demais funcionalidades, neste caso, basta aperta qualquer tecla, como a letra “q” e esperar o término apertando ENTER e o Node.js estará corretamente instalado.

- Com os Vs-Code e Node.js instalados, vamos instalar os pacotes que vão abrigar o robô. Neste projeto nós vamos utilizar o pacote gratuito do Pedro Lopez, um programador das Ilhas do Caribe muito fera. O whatsapp-web.js, guarde bem esse nome pois esta biblioteca abençoada vai nos entregar o **OURO**.

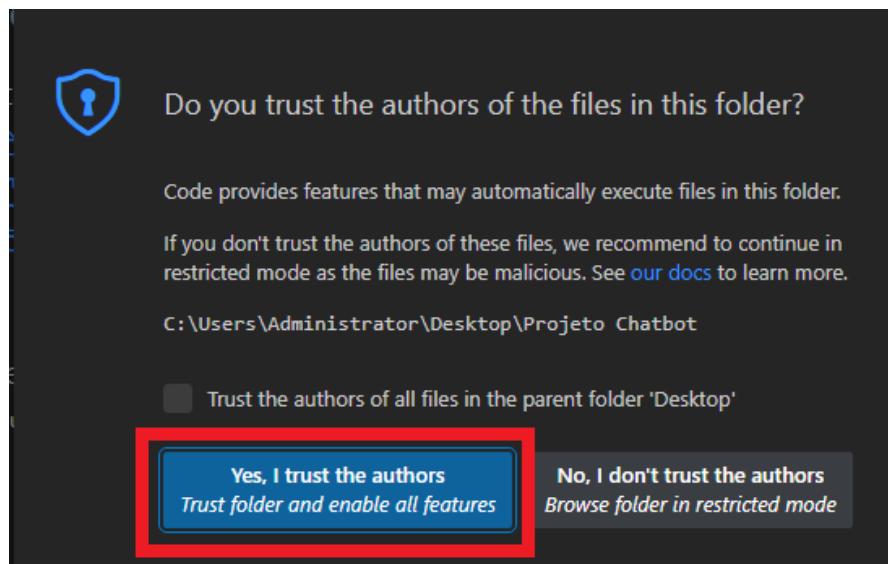
- Após a instalação do Node.js é recomendado fechar e abrir novamente o seu VS-Code caso ele esteja funcionando
- Com o VS-Code aberto, vamos adicionar a pasta em que vamos trabalhar, dê preferência para aquela pasta que você criou no início. Para isso clique em “File” ou “Arquivo” no canto superior esquerdo e escolhemos a opção “Open Folder” ou “Abrir Pasta” como indicado na figura abaixo:



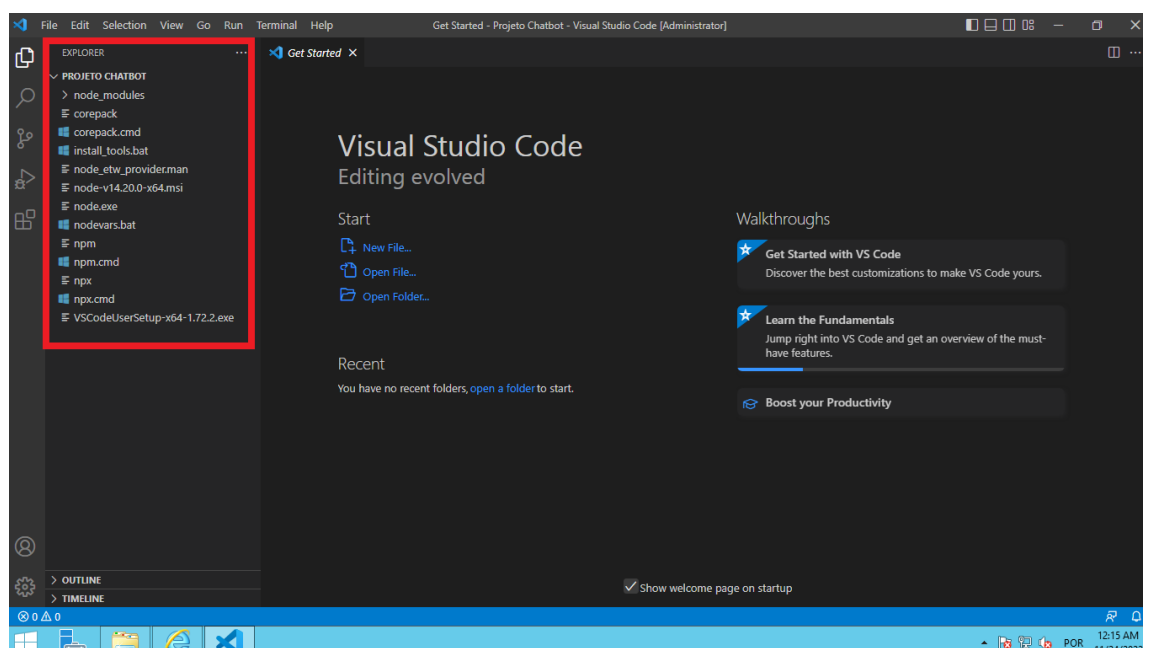
Selecionamos a pasta que criamos para o projeto e clicamos em “Select Folder” ou “Selecionar Pasta”:



O Vs-Code pode perguntar se você confia na pasta e em seus conteúdo para trabalhar o projeto. Basta clicar que sim como na figura:



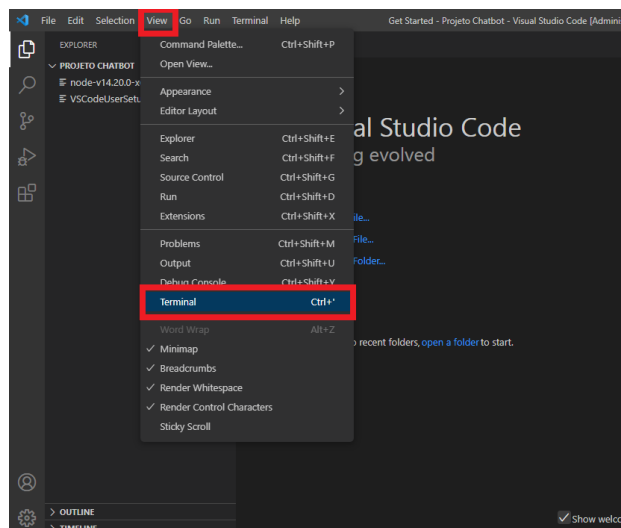
- Pronto! Carregamos a pasta que usaremos e sua tela ficará assim:



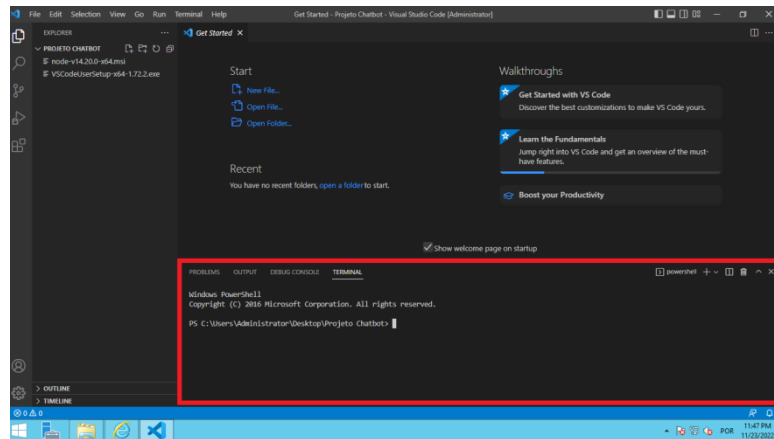
Note que os conteúdos da pasta vão aparecer no canto esquerdo

destacado, você poderá navegar pelo conteúdo da pasta, criar e modificar arquivos por aí. Fazendo assim fica mais rápido e intuitivo!

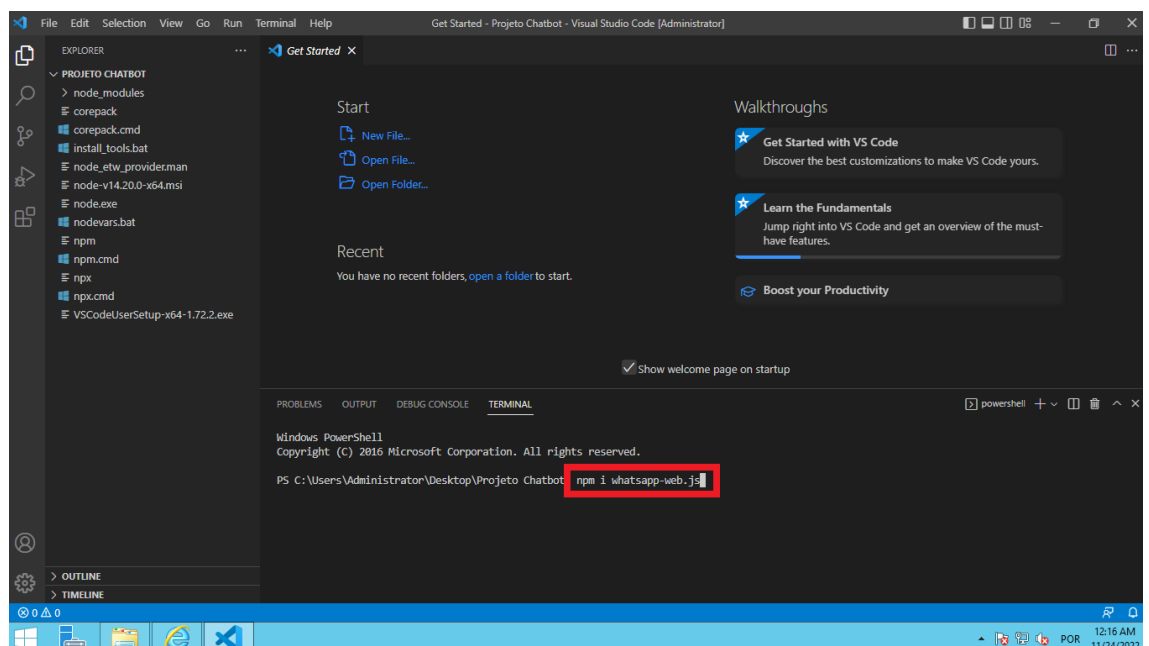
- Agora com tudo pronto, vamos terminar de instalar os pacotes restantes, a abençoada biblioteca do Pedro Lopez com os recursos para Whatsapp, uma atualização dela que nos permitirá enviar botões e listas e uma biblioteca que permite a confecção de qr-code. Bora lá!
- E para isso vamos carregar o terminal do VS-Code onde iremos inserir os comandos para baixar pacotes, carregar o robô, ler qr-code, etc. Clique em “View” ou “Ver” e depois em “Terminal” como destacado na figura:



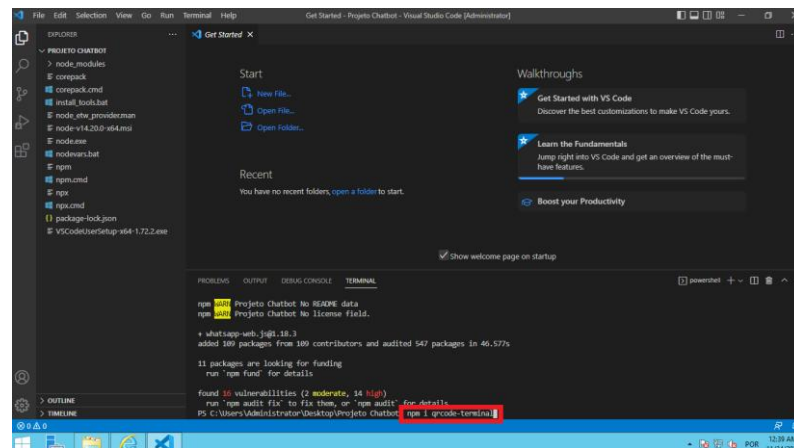
Aguarde um pouco até o terminal carregar por completo e você terá um prompt pronto para receber suas instruções na parte inferior da tela como mostra a figura:



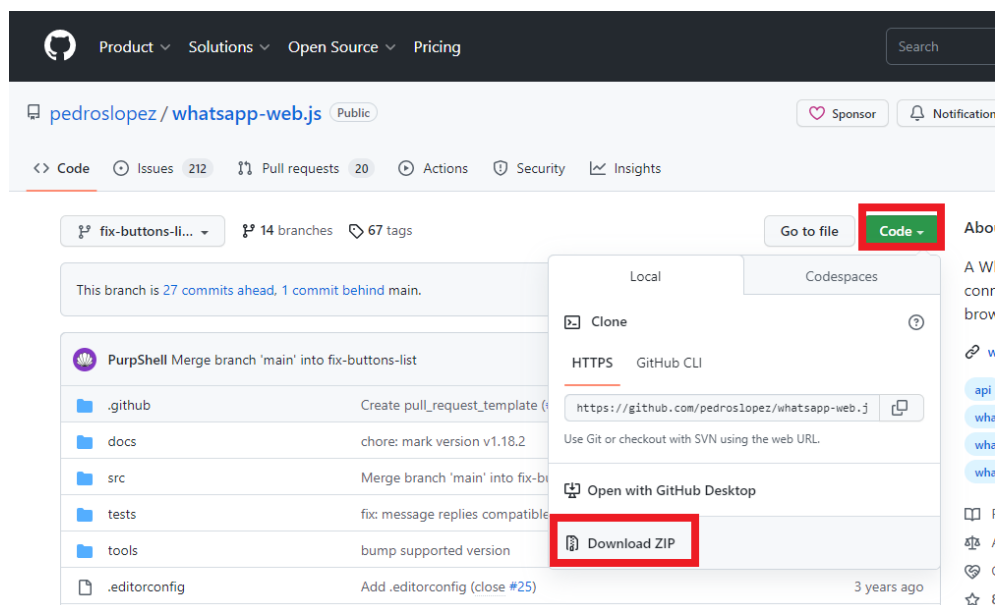
- Com tudo pronto vamos proceder com a instalação da biblioteca abençoada do Pedro Lopez e para isso clicamos no prompt do Terminal que acabamos de carregar e digitamos **"npm i whatsapp-web.js"** e apertamos ENTER como na imagem abaixo:



Aguardamos a instalação da biblioteca do Pedro Lopez terminar e partimos para a instalação da biblioteca de geração de qr-code. Essa biblioteca usaremos para criar a ponte que nos permitirá conectar nosso Whatsapp ao robô. Para isso no mesmo terminal digitamos **"npm i qrcode-terminal"** e apertamos ENTER como na imagem abaixo.



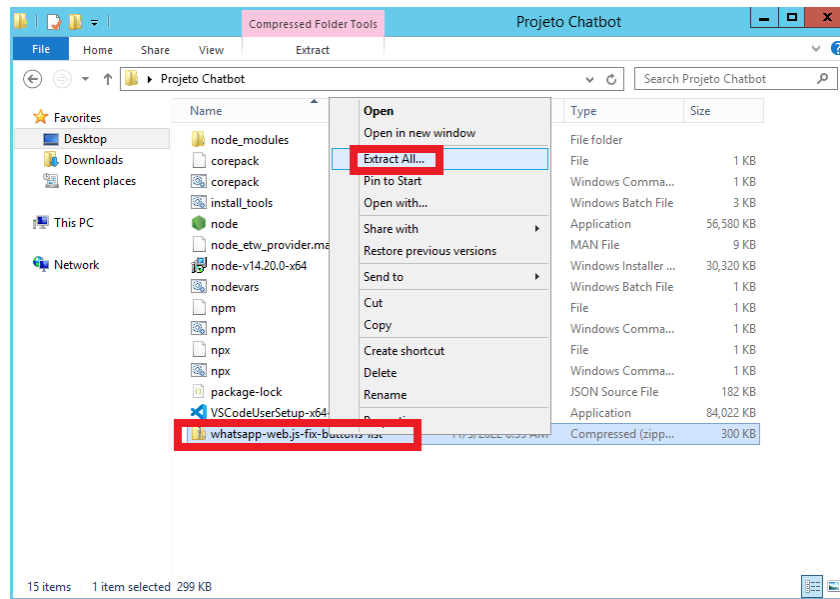
- Com o pacote qrcode-terminal instalado, estamos quase prontos com a rodada de instalações. Precisamos apenas atualizar a biblioteca do Pedro Lopez por uma versão com os botões e listas atualizada. **Aqui é importante notar que por ser uma biblioteca aberta e contributiva, novas funcionalidades e versões são adicionadas com frequência.** Neste passo vamos carregar uma versão com os botões prontos e para isso vamos acessar o link: <https://github.com/pedroslopes/whatsapp-web.js/tree/fix-buttons-list> E clicaremos em “Code” e depois em “Download ZIP” como na figura destacada:



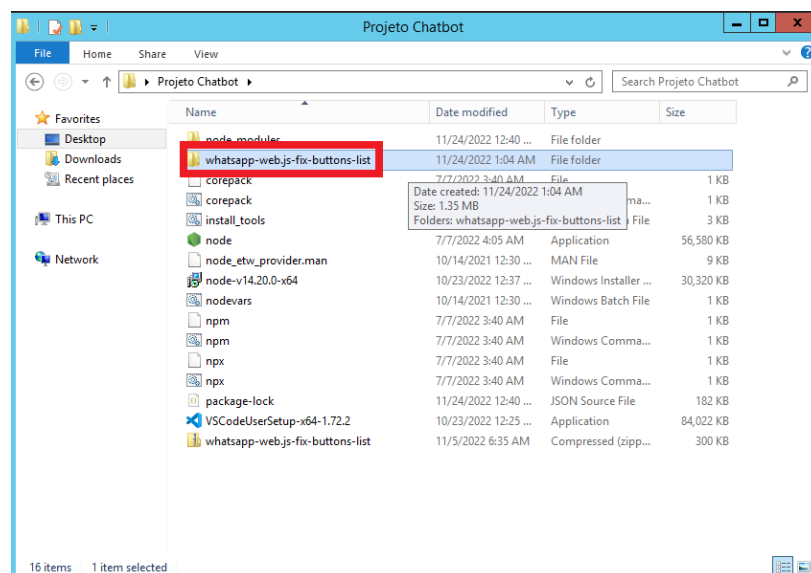
Isso vai baixar uma pasta compactada chamada “whatsapp-web.js-fix-buttons-list” e você vai também coloca-la na pasta do projeto e



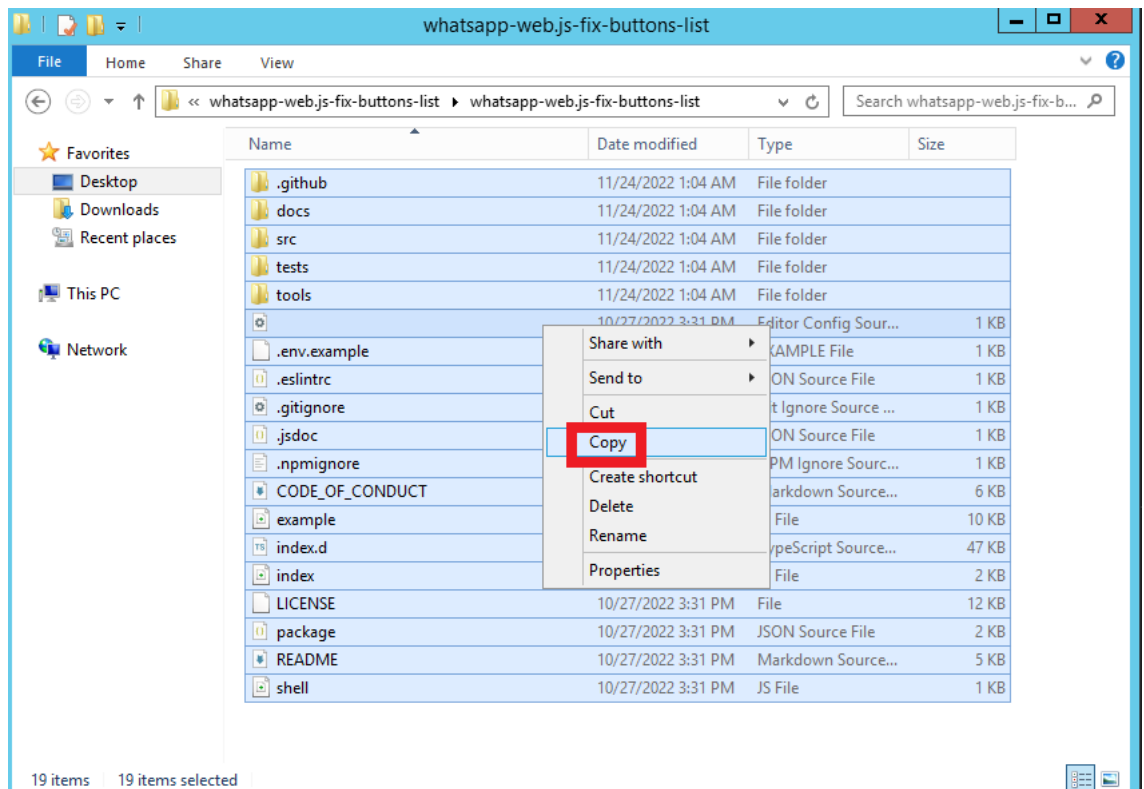
fazer a sua descompactação como na figura abaixo:



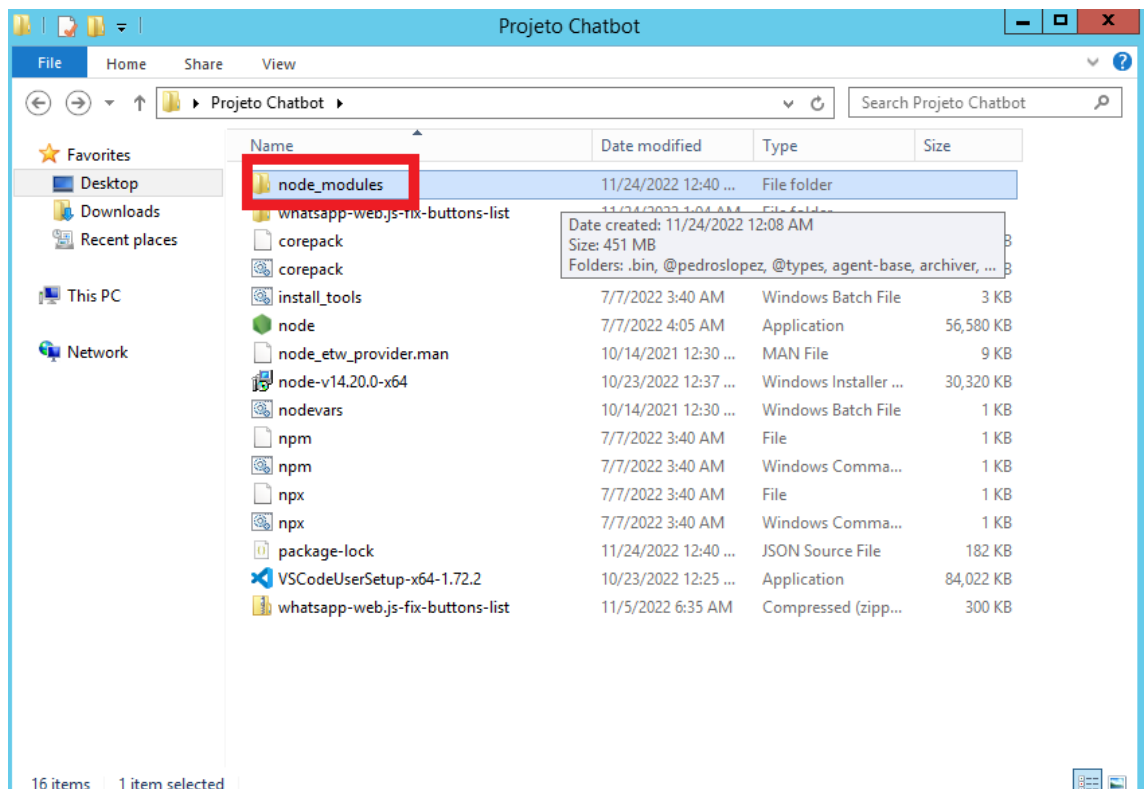
Após a extração você vai notar uma nova pasta com o nome do arquivo compactado. O conteúdo desta pasta contém o projeto do Pedro Lopez com a funcionalidade de botões e listas configurada.



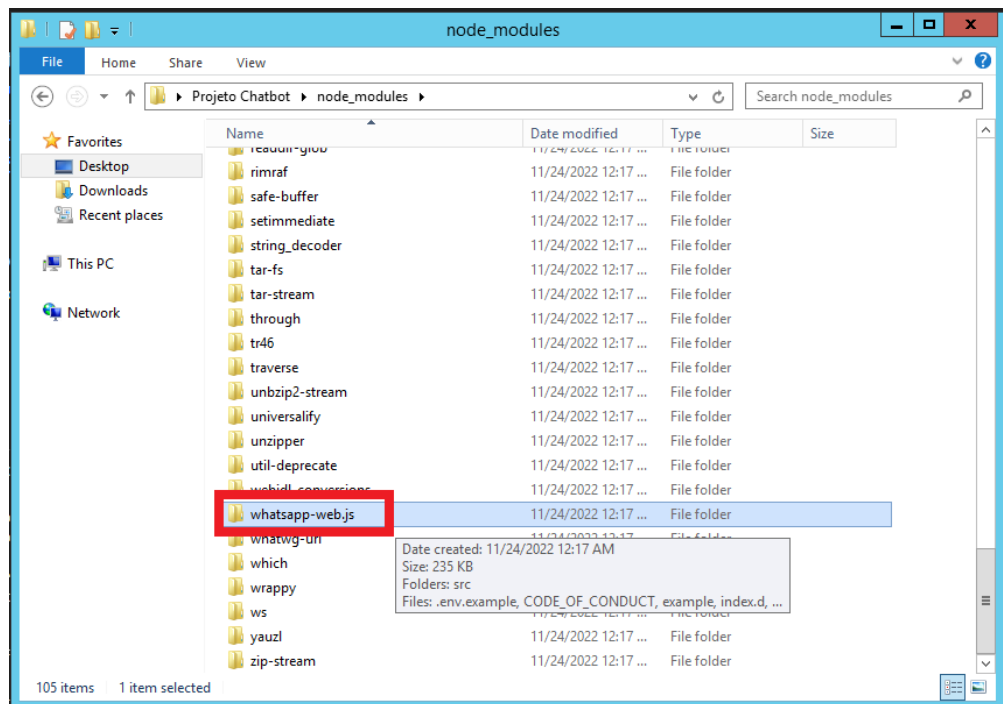
- Vamos substituir os arquivos da pasta da nossa biblioteca pelos arquivos da nova biblioteca com os botões arrumados e para isso abrimos a nova pasta destacada acima, selecionamos todo seu conteúdo e copiamos com Ctrl+C



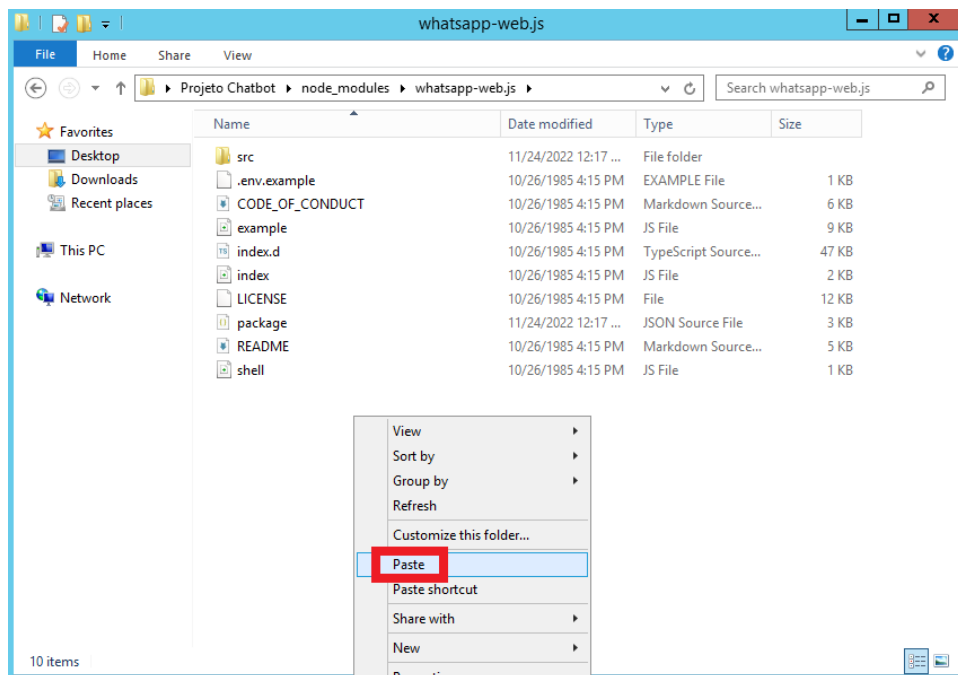
Então vamos até a pasta da biblioteca original e colamos lá todo o conteúdo, para isso voltamos na pasta do projeto e clicamos na pasta “node\_modules”



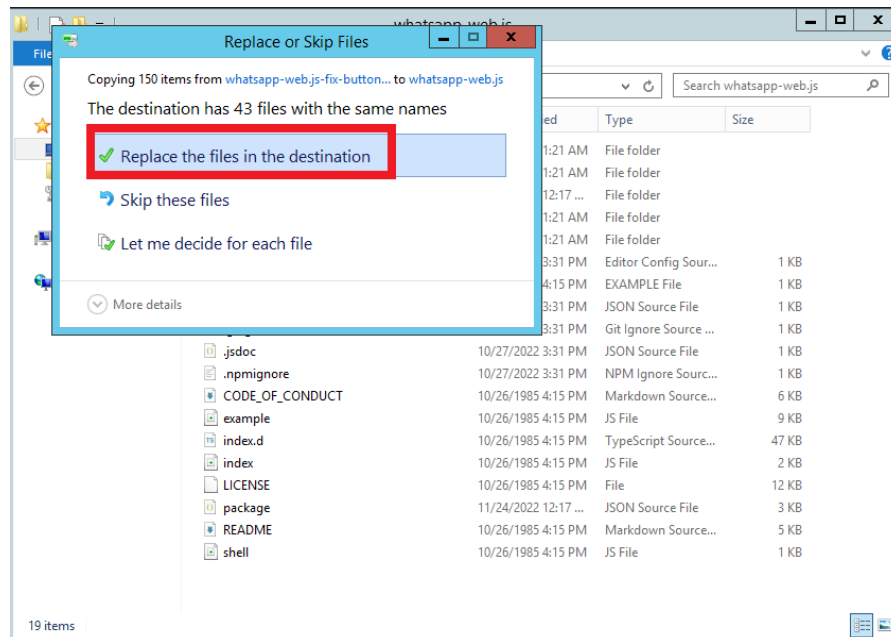
E procuramos pela pasta “whatsapp-web.js” para acessarmos seu conteúdo com dois cliques:



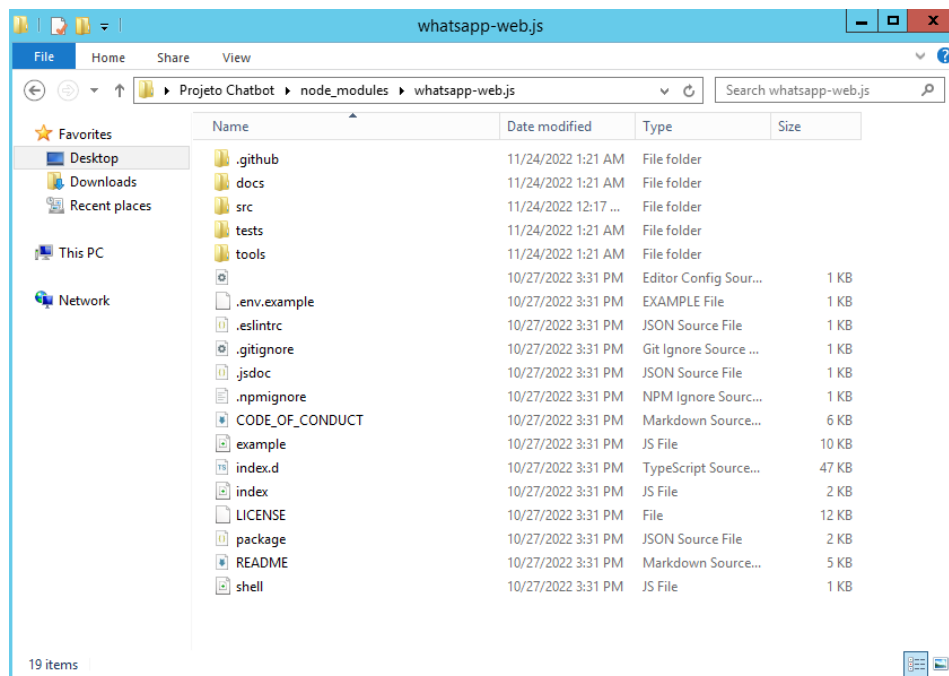
Colamos o conteúdo com Ctrl+V ou botão direito do mouse e “Colar” ou “Paste” como na figura:



Em seguida confirme todas as substituições:



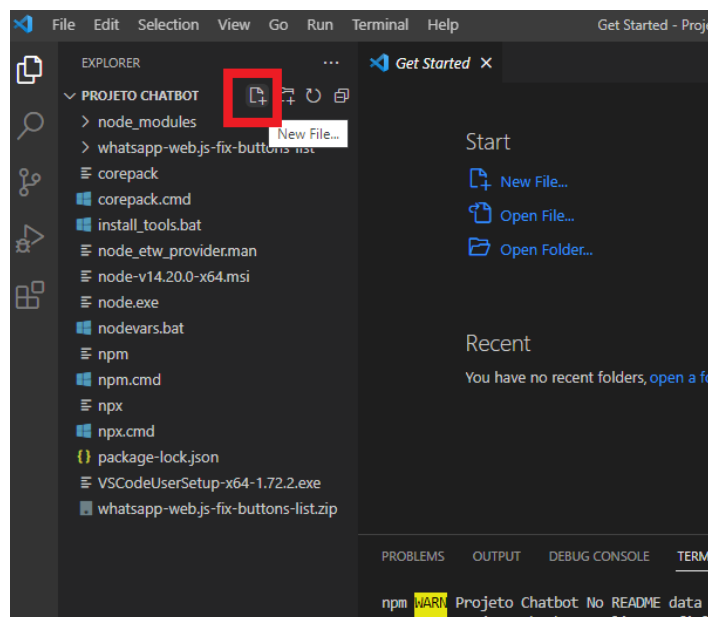
E... pronto. Nossa biblioteca está habilitada para trabalhar com botões e listas!



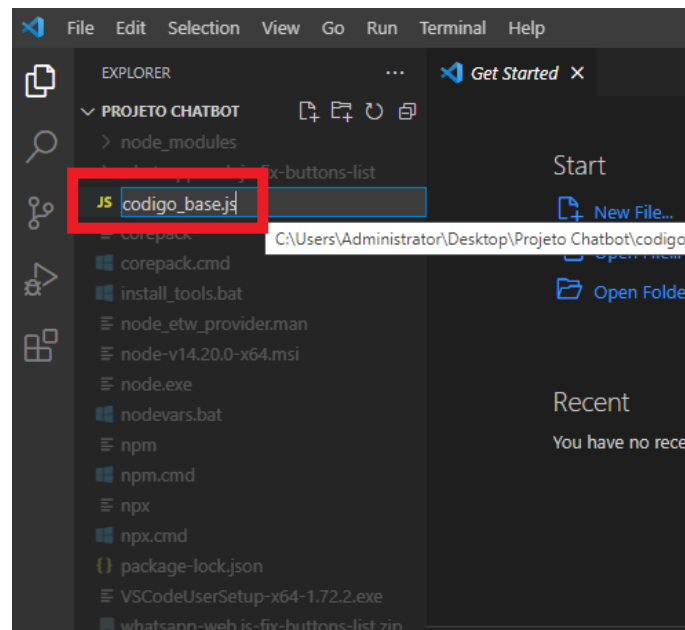
- Finalmente está tudo pronto, nosso trabalho em breve será recompensado e o processo de instalação só precisa ser feito uma única vez.

# Modulo Practice – Preparando o terreno para rodar nosso código base

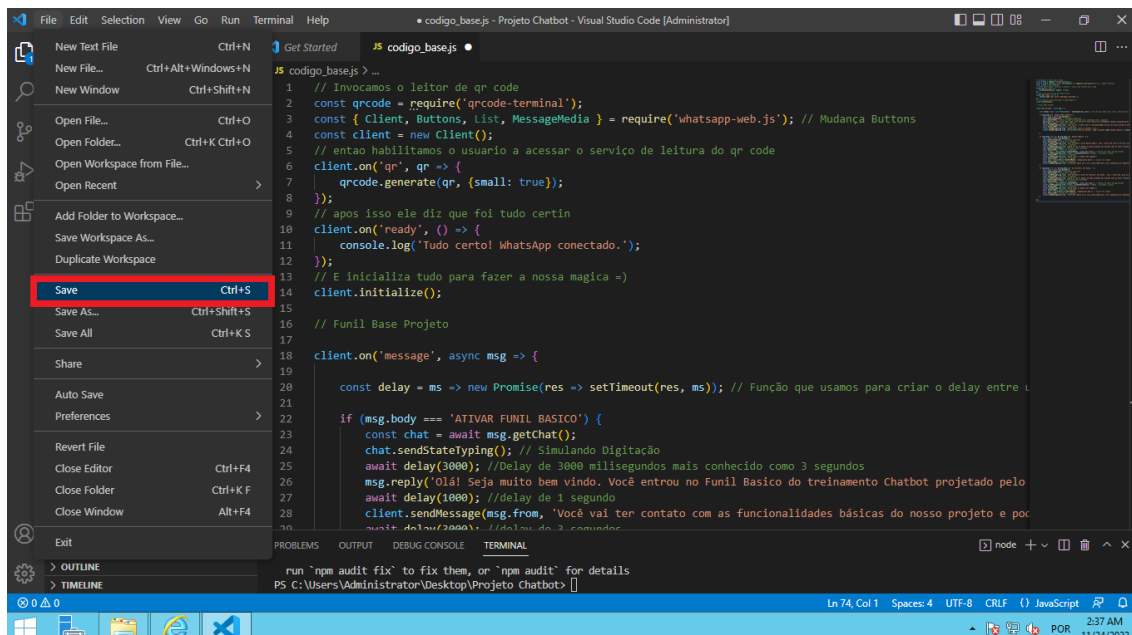
- Com toda a instalação completa, vamos carregar e executar o primeiro script com um funil base de atendimento. O objetivo é te mostrar como carregar os funis com um script base de exemplo que vai conter as funcionalidades básicas da nossa ferramenta, assim você estará apto a rodar seus próprios funis. O funil base contém a possibilidade de enviar áudios como se fossem gravados, imagens, botões, listas, delay e simulação de texto digitando e áudio gravando.
- Vamos buscar o “codigo\_base.js” que está na área de membros do seu produto e arrastar para a pasta do projeto ou você pode copiar e colar o código base que está nos anexos deste Guia Incrível num novo arquivo .js. Para isso, nós clicamos no ícone “Novo arquivo” no VS-Code e o nomeamos como “código\_base.js”, então basta colar o código que está no anexo deste guia. Acompanhe as imagens abaixo:



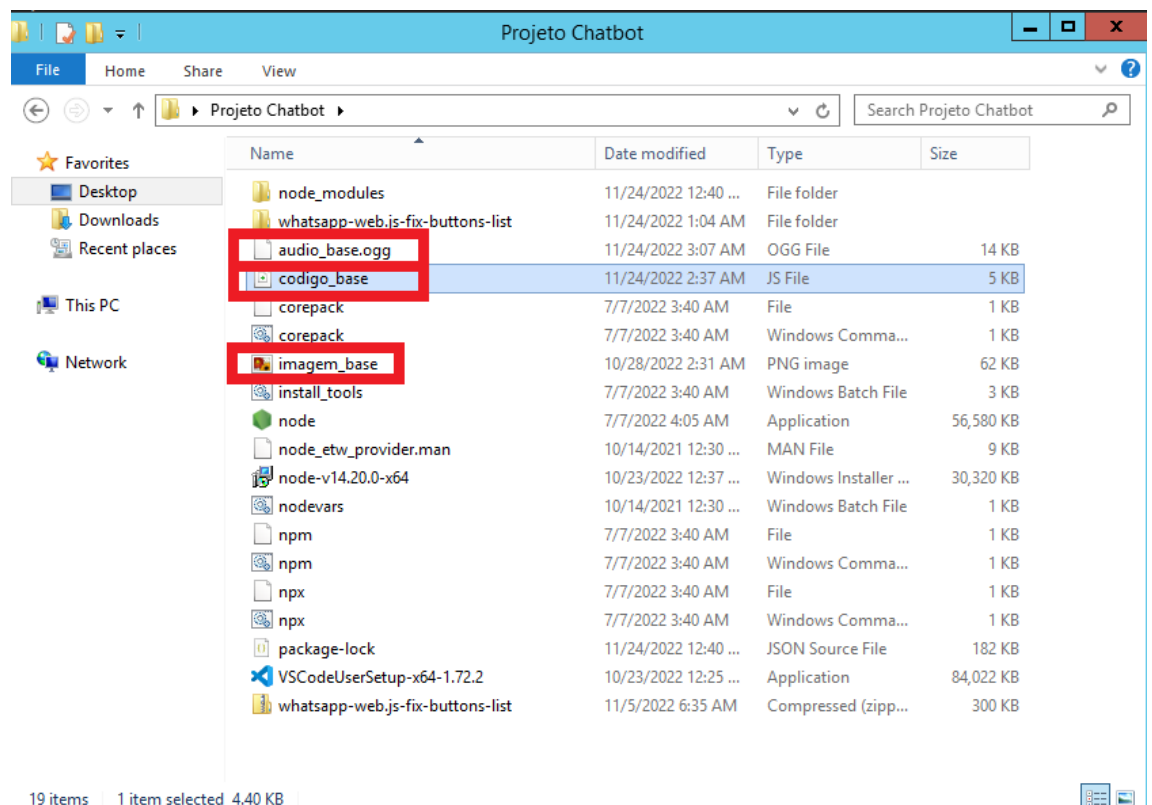
Após clicar em “New File” ou “Novo Arquivo” você vai nomear para “código\_base.js” como na imagem:



E após confirmar com ENTER, o arquivo se abrirá como uma aba em seu VS-Code. É nesta aba que você colará o código base e vai salvar clicando em “File” e depois em “Save”:



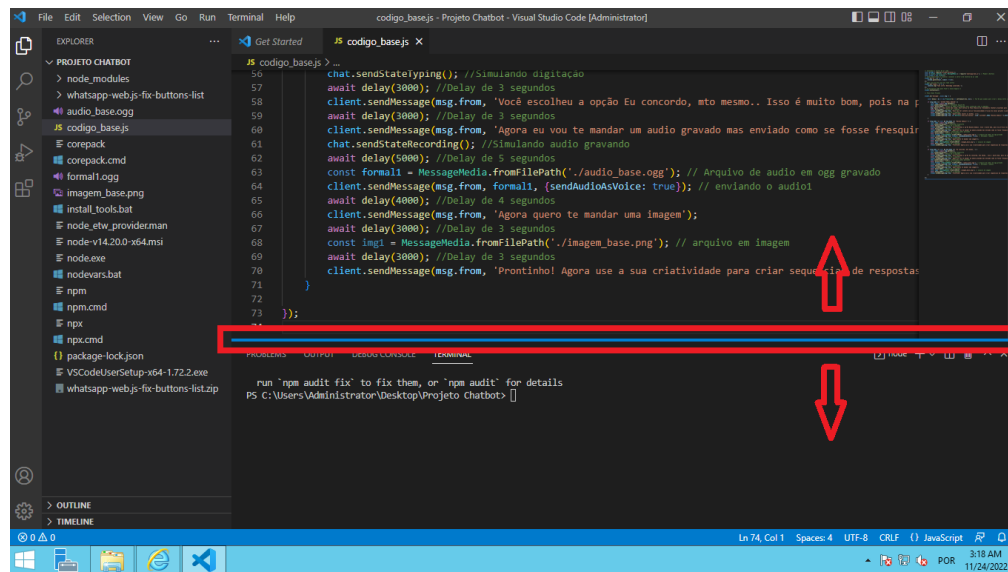
- Lembrando que você pode arrastar o arquivo “codigo\_base.js” pronto para a sua pasta do projeto e abri-lo em seu VS-Code com dois cliques no navegador do lado esquerdo. O código base está dentro da sua área de membros.
- Os arquivos “audio\_base.ogg” e “imagem\_base.png” serão usados no teste e são um áudio de whatsapp e uma imagem que preparei. Seus funis personalizados serão feitos exatamente desta maneira, você vai gravar os áudios no seu whatsapp, baixa-los e deixa-los na pasta do projeto. Assim você fará com esses dois arquivos testes, eles também estão na área de membros junto ao código base. Você vai adicionar todos eles na pasta do projeto como na figura abaixo:



- Com o código base pronto, vamos enfim colocar para rodar o primeiro projeto de chatbot no whatsapp!

# Modulo Running – Rodando e testando o código base

- No terminal, arrastamos para cima o seu tamanho, para ganharmos um espaço para a leitura do qr code que vamos gerar. A imagem abaixo:



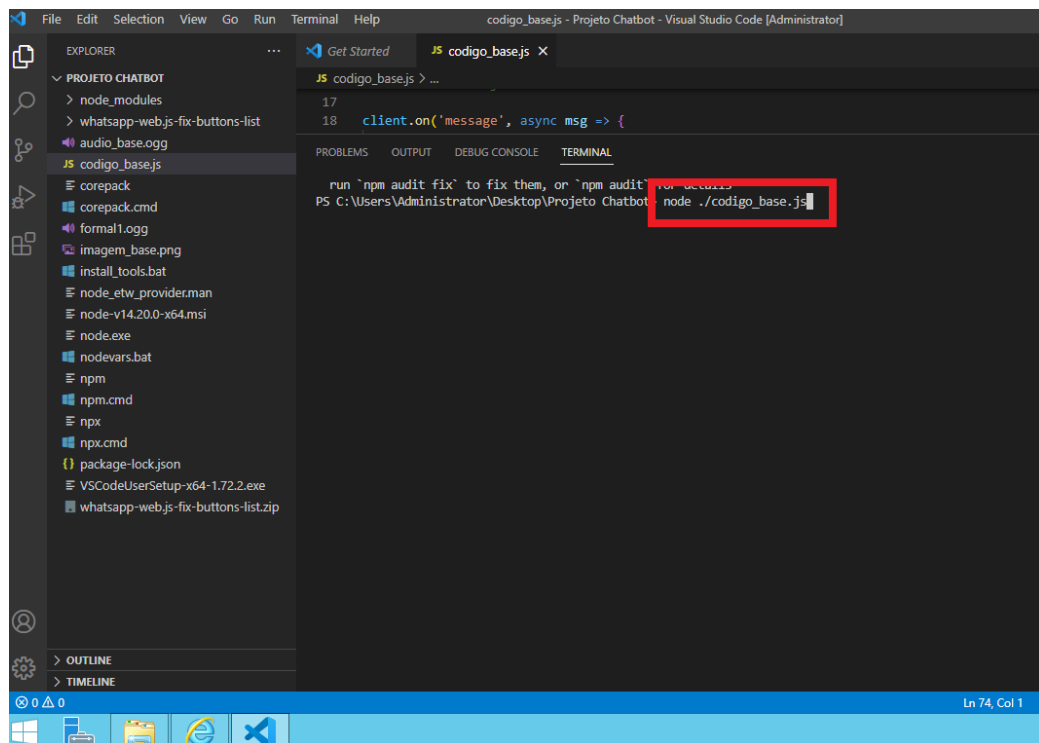
Com um espaço maior buscado, clicamos no console e digitamos

**`“node ./codigo_base.js”`**

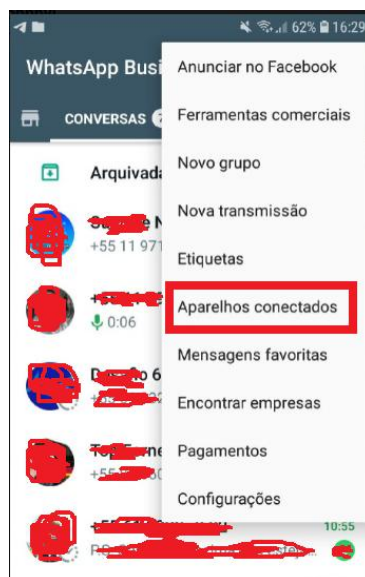
Isso fará o código base do funil ser executado após apertarmos

ENTER:





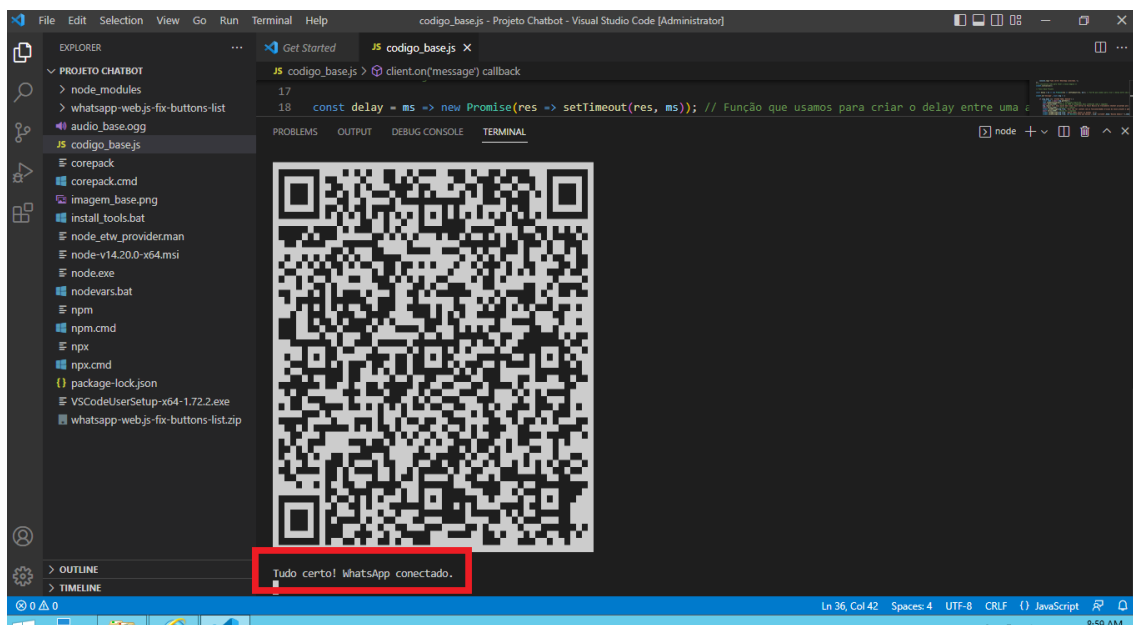
Será gerado um código qr na tela do terminal, este código você irá fazer a leitura com o seu whatsapp normal ou business para habilitar o robô. É como se você estivesse logando num whatsapp web, então basta ir em **aparelhos conectados** em seu aplicativo e realizar a leitura do qr-code.



Clicamos em Conectar Aparelho:



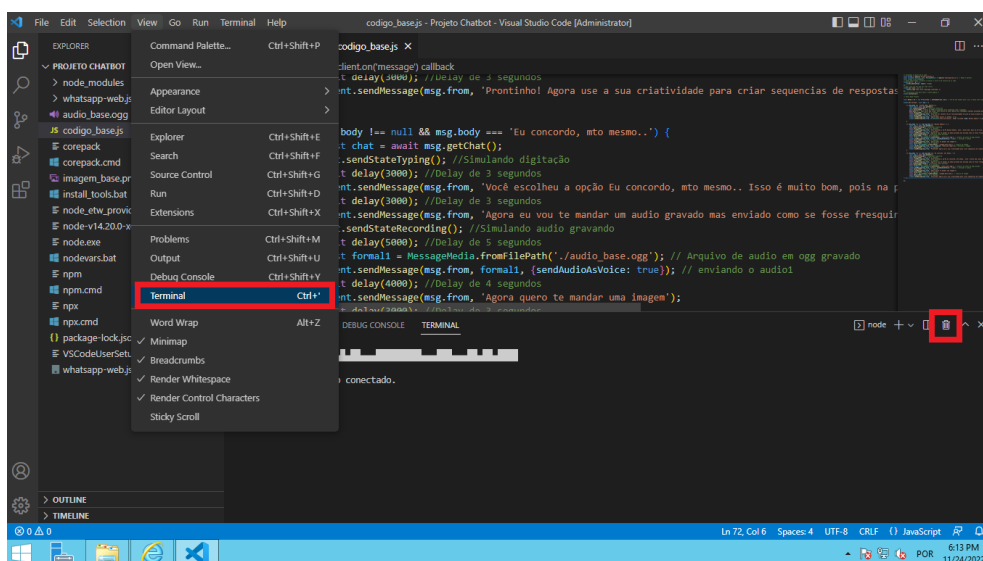
Então procedemos com a leitura do qr-code gerado no terminal do VS-Code com o nosso celular apontado para a tela até conseguirmos logar:



A mensagem “Tudo certo! Whatsapp conectado” indica que está tudo pronto e podemos interagir com o robô enviando “ATIVAR FUNIL BASICO”.

# Modulo Customizing – Entendendo os blocos de interação e criando os seus próprios.

- Agora que já temos todos os programas instalados, vamos estudar o significado dos blocos de interação e como criar os seus próprios. Mas antes preciso te dar algumas instruções de quem aplica essa técnica todos os dias.
- Ao subir um novo funil, ou atualizar o antigo você precisará parar de rodar o funil atual deslogando do seu celular na aba de Aparelhos Conectados clicando sobre o nome que representa o acesso do robô e depois em “Remover aparelho”. No Whatsapp Business você clicará nos três pontinhos e a opção “Remover Aparelho” irá aparecer. Na maioria das vezes o terminal do VS-Code liberar para inserir o novo comando, mas se demorar você pode clicar no ícone de uma Lixeira para fechar o terminal e depois abri-lo novamente clicando em “View” ou “Ver” e depois em “Terminal” como já estudamos e um novo terminal irá se abrir.



Então rode o novo funil com o comando já conhecido

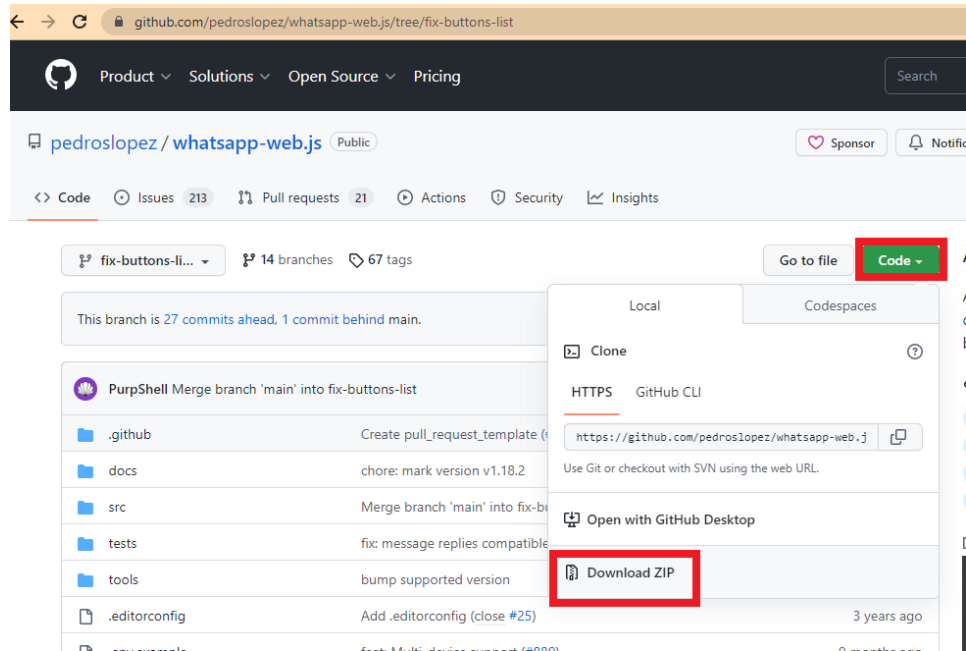
`“node ./seu_novo_funil.js”`

Recomendo que faça esse procedimento no final da noite, quando poucas pessoas estão interagindo com o robô. Já deixe o novo arquivo na pasta do projeto pra ganhar tempo. É possível realizar a troca em menos de 2 minutos, mas é bom garantir a qualidade da sua operação tomando esses cuidados.

- Também é importante dizer que de tempos em tempos a biblioteca que utilizamos atualiza e quando uma atualização mais forte entre em vigor, a biblioteca antiga onde estamos pode não carregar o robô. Nesse caso, você verá uma mensagem de erro no Terminal. Para normalizar basta resetar o Terminal como no tópico anterior, ir na pagina da biblioteca que já estudamos e você pode entrar por este link:

<https://github.com/pedroslopez/whatsapp-web.js/tree/fix-buttons-list>

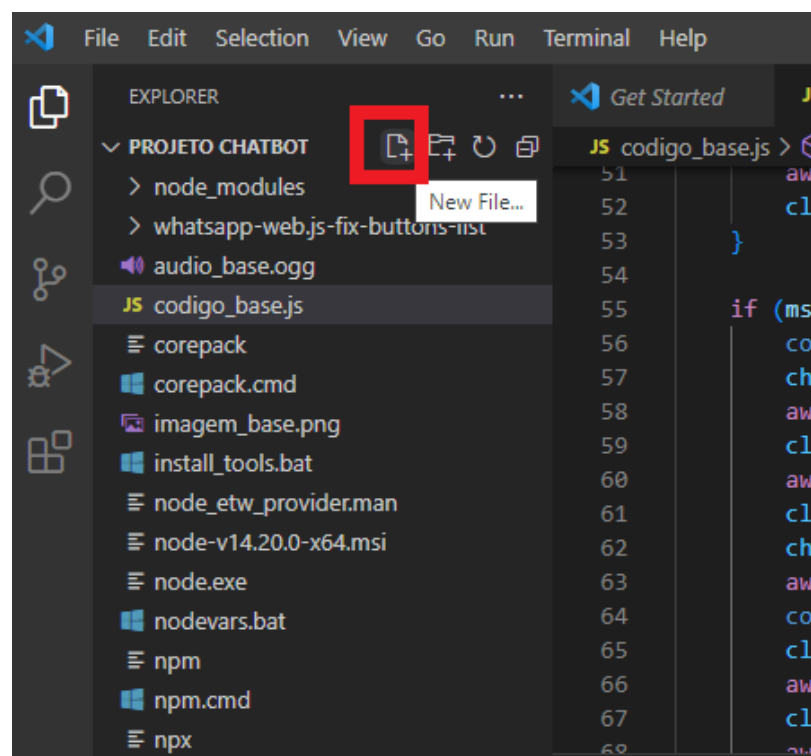
baixar o arquivo compactado da biblioteca atualizado:



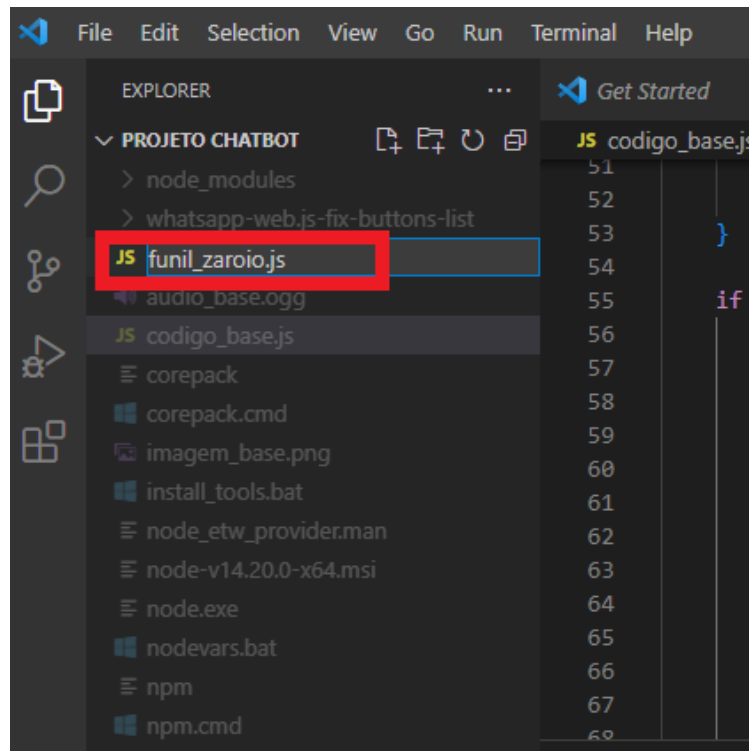
E então colando e substituindo o conteúdo da nova pasta baixada na pasta “whatsapp-web.js” dentro da pasta “node\_modules” e confirmar a substituição do mesmo jeito que estudamos quando estávamos a instalar

a versão com botões. As atualizações dão esse pequeno trabalho extra ao trocar de funil, mas elas costumam entregar novidades boas então eu recomendo que não apenas faça esse procedimento quando trocar de funil e tomar algum erro, mas de forma mais periódica. Uma vez por semana é suficiente.

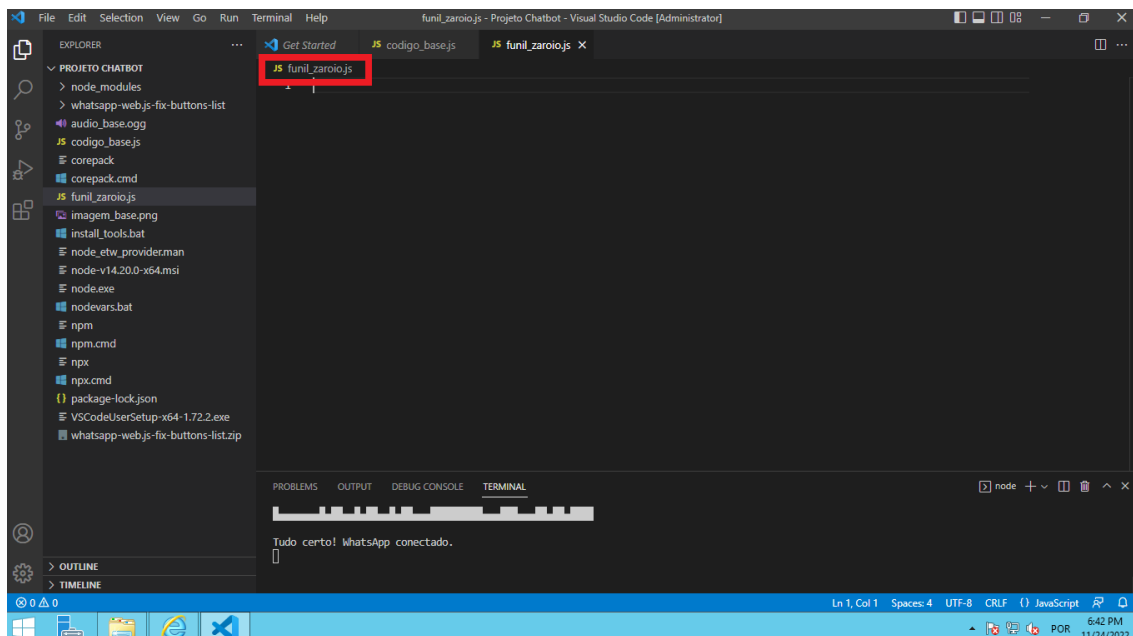
- Agora vamos estudar a estrutura dos blocos.
- Todo script ou funil de atendimento deve conter um final “.js” que você abrirá na aba do VS-Code como estudamos anteriormente. Você pode criar um arquivo em branco e ir adicionando os códigos de outros arquivos ou pode copiar e colar um funil já existente, renomear e editar no VS-Code como estudamos anteriormente. O caminho fica a seu critério. Neste passo vou fazer o caminho construindo do zero. Vamos lá!
- No VS-Code clique “New File” ou “Novo Arquivo” como mostrado na figura:



Escolhi o nome “funil\_zaroio.js”, note que a “.js” está lá sempre presente:



Abre-se uma nova aba sem nada escrito:



É nessa nova aba que vamos digitar o código do novo projeto. O primeiro bloco de código será igual para todos os projetos, pois ele lida com a parte de acesso e leitura do whatsapp, então apenas copie e cole o bloco abaixo:

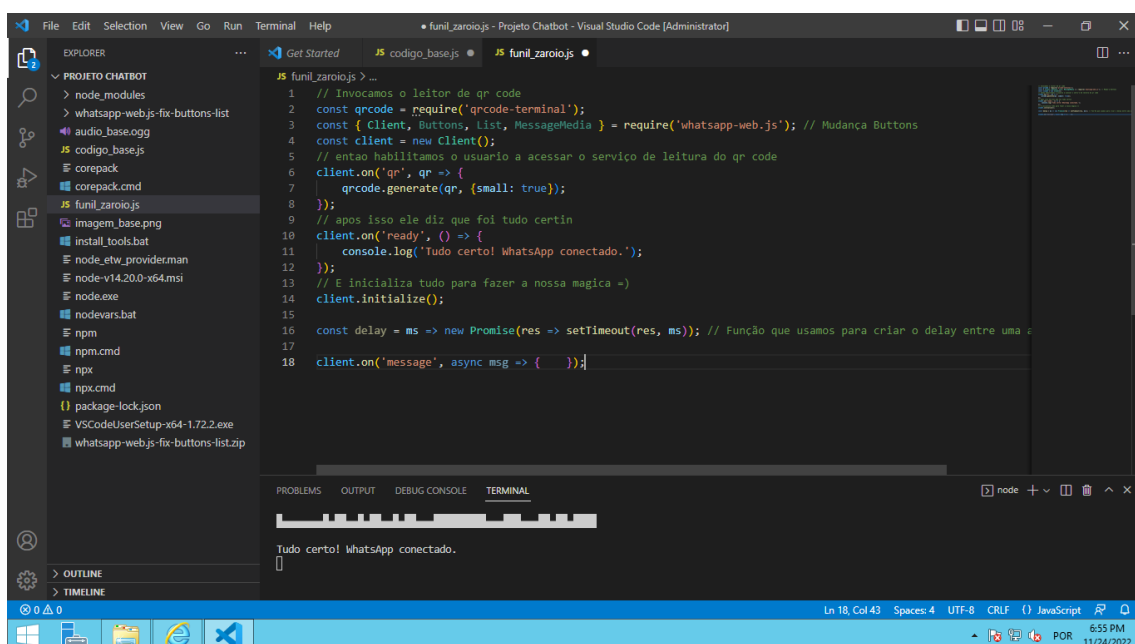
```
// Invocamos o leitor de qr code
const qrcode = require('qrcode-terminal');
const { Client, Buttons, List, MessageMedia } = require('whatsapp-web.js'); // Mudança Buttons
const client = new Client();
// entao habilitamos o usuario a acessar o serviço de leitura do qr code
client.on('qr', qr => {
    qrcode.generate(qr, {small: true});
});
// apos isso ele diz que foi tudo certin
client.on('ready', () => {
    console.log('Tudo certo! WhatsApp conectado.');
```

```
});
// E inicializa tudo para fazer a nossa magia =)
client.initialize();

const delay = ms => new Promise(res => setTimeout(res, ms)); // Função
que usamos para criar o delay entre uma ação e outra

client.on('message', async msg => {    });
```

E sua aba do “funil\_zaroio.js” ficará assim:



- Agora vamos inserir os blocos de interação!
- Todos estes blocos ficarão dentro desta estrutura no espaço marcado em amarelo:

```
client.on('message', async msg => {---});
```

Um bloco de interação é uma ação do robô que será executada quando qualquer pessoa do seu Whatsapp digitar uma sequência de letras determinadas. O primeiro bloco eu gosto de fazer como um bloco de gatilho, ou seja, é o primeiro bloco que a pessoa vai receber quando nos chamar no Whatsapp. Vamos ao exemplo:

```
if (msg.body === 'OLHA O FUNIL ZAROIIO') {  
  
}
```

Neste caso, um bloco vai se ativar quando alguém te mandar uma mensagem contendo “OLHA O FUNIL ZAROIIO” exatamente com essas letras, sem acento e maiúsculo. É importante dizer isso pois letras minúsculas são vistas como letras diferentes de maiúsculas, ou seja, G é diferente de g por exemplo. Como não existe nenhuma ação dentro bloco, ele vai se ativar mas não fará nada. Vamos agora estudar as principais ações que podemos colocar dentro do bloco.

Seu código do funil zaroio deve estar com essa cara:

```
JS funil_zaroio.js > ...  
1 // Invocamos o leitor de qr code  
2 const qrcode = require('qrcode-terminal');  
3 const { Client, Buttons, List, MessageMedia } = require('whatsapp-web.js'); // Mudança Buttons  
4 const client = new Client();  
5 // entao habilitamos o usuario a acessar o serviço de leitura do qr code  
6 client.on('qr', qr => {  
7   qrcode.generate(qr, {small: true});  
8 });  
9 // apos isso ele diz que foi tudo certin  
10 client.on('ready', () => {  
11   console.log('Tudo certo! WhatsApp conectado.');12 });  
13 // E inicializa tudo para fazer a nossa magica =)  
14 client.initialize();  
15  
16 const delay = ms => new Promise(res => setTimeout(res, ms)); // Função que usamos para criar o delay entre uma  
17  
18 client.on('message', async msg => {  
19   if (msg.body === 'OLHA O FUNIL ZAROIIO') {  
20  
21   }  
22 });
```



As funcionalidades fundamentais são simples de entender e agora vou descrever cada uma delas e depois implementar em nosso funil zaroio.

## Enviando Mensagem de Texto

```
msg.reply('Olá! Seja muito bem vindo. Você entrou no Funil Basico do treinamento Chatbot projetado pelo Johnny'); //Primeira mensagem de texto
```

O código “`msg.reply('Seu texto aqui');`” envia um texto que estará entre as aspas. Você pode escrever absolutamente qualquer coisa, poderá colar emoticons e texto em negrito. Este trecho vai conter a sua copy escrita propriamente dita. Se você escrever outro `msg.reply()` depois, o robô vai mandar outra mensagem em separado.

## Simulando “Digitando Texto”

```
const chat = await msg.getChat();  
chat.sendStateTyping(); //Simulando digitação
```

Este código vai simular aqueles estados de “digitando texto” que aparecem quando um usuário está escrevendo. Coloque esta simulação no topo do bloco para uma vibe de resposta realista ao seu cliente.

## Enviando Audio Como se Fosse Gravado na Hora

```
const formal1 = MessageMedia.fromFilePath('./audio_base.ogg'); // Arquivo de audio em ogg gravado  
client.sendMessage(msg.from, formal1, {sendAudioAsVoice: true});  
// enviando o audio1
```

Este trecho quando colocado dentro do bloco, envia um áudio como se fosse gravado na hora. Vamos notar o nome “`formal1`” que pode ser mudado, ele é apenas uma variável criada para guardar o arquivo com o áudio. Recomendo que este arquivo com áudio seja gravado pelo próprio whatsapp e baixado para a pasta do projeto, você pode fazer isso usando o Whatsapp web para baixar. O arquivo de áudio quando baixado vem na extensão `.ogg` e no exemplo está marcado como “`audio_base.ogg`” mas você pode nomea-lo como preferir.

## Simulando Gravando Audio

```
chat.sendStateRecording(); //Simulando audio gravando
```

Funcionalidade parecida com o “Digitando texto” mas para áudio. O trecho “`const chat = await msg.getChat();`” é necessário estar no topo do bloco caso não tenha adicionado.

## Enviando Imagem

```
const img1 = MessageMedia.fromFilePath('./imagem_base.png'); // arquivo em imagem
client.sendMessage(msg.from, img1, {caption: 'Olha que legal'});
//Enviando a imagem
```

A imagem deve estar em png e no exemplo do código a variável `img1` guardará o arquivo `imagem_base.png` que deverá estar na sua pasta do projeto. Estes nomes podem estar como você preferir e você pode mandar imagens em sequencia bastando apenas carrega-las e envia-las como preferir.

## Inserindo Delay

```
await delay(1000); //delay de 1 segundo
```

Um delay serve para fazer o robô aguardar alguns segundos para que as mensagens não fiquem tão rápidas aos nossos clientes e eles tenham tempo para consumi-las antes das proximas mensagens. No caso do exemplo acima, o robô vai aguardar 1000 milisegundos, ou 1 segundo até executar a próxima linha dentro do bloco.

## Enviando Botões

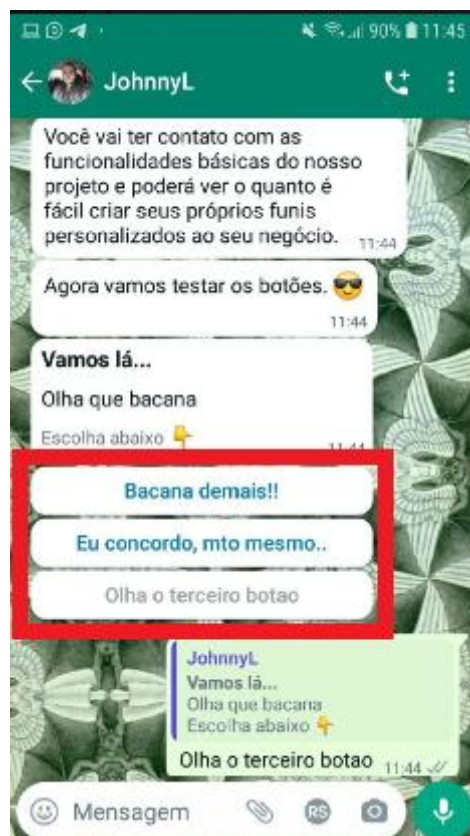
```
client.sendMessage(msg.from, new Buttons('Olha que bacana',
[{'id': 'customId', body: 'Bacana demais!!'}, {'body: 'Eu concordo, mto mesmo..'}], 'Vamos lá...', 'Escolha abaixo 🗳️'));
```

Enfim chegamos nos botões! Este trecho vai enviar um botão para o seu cliente escolher uma resposta dentre as possíveis. Neste exemplo existirão

dois botões, um com “Bacana demais!!” e outro com “Eu concordo, mto mesmo..”. Quando o cliente escolher um dos botões, será como se ele tivesse escrito exatamente a resposta e você poderá criar um bloco pronto para responder caso a tal resposta seja recebida. Na versão atual da biblioteca é possível enviar até 5 botões e disso você usará a sua imaginação para criar poderosos funis de atendimento. Toda vez que você adicionar um novo botão, deve terminar o ultimo “{” com “,” iniciar com um novo “{body: ‘Novo texto’}” veja no exemplo abaixo como ficaria o mesmo código anterior com um botão adicional:

```
client.sendMessage(msg.from, new Buttons('Olha que bacana',  
[{'id':'customId',body:'Bacana demais!!'},{'body':'Eu concordo, mto  
mesmo..'}, {'body':'Olha o terceiro botao'}], 'Vamos lá...', 'Escolha  
abaixo 📌'));
```

O código acima nos permite criar esses três botões como destacados na imagem onde os botões foram enviados e o usuário apertou no terceiro botão:



Agora que te apresentei aos elementos básicos, já estamos prontos para unir todos eles em nossos blocos de códigos que vão conversar entre si. Bora lá!

- Com as funcionalidades descritas já conseguimos montar o conteúdo do primeiro bloco:

```
if (msg.body === 'OLHA O FUNIL ZAROIO') {  
  const chat = await msg.getChat();  
  chat.sendStateTyping(); // Simulando Digitação  
  await delay(3000); //Delay de 3000 milisegundos mais conhecido  
  como 3 segundos  
  msg.reply('Eu sou o Funil Zaroio'); //Primeira mensagem de texto  
  await delay(1000); //delay de 1 segundo  
  client.sendMessage(msg.from, 'Esta é a segunda mensagem do funil  
  zaroio');  
  await delay(3000); //delay de 3 segundos  
  client.sendMessage(msg.from, 'Olha os 3 botoes zaroio');  
  client.sendMessage(msg.from, new Buttons('Olha que bacana',  
  [{id:'customId',body:'Zaroio 1'}, {body:'Zaroio 2'}, {body:'Zaroio 3'}],  
  'Vamos lá...', 'Escolha abaixo 🗳️'));  
}
```

- Este bloco de instruções vai primeiramente ser ativado quando seu cliente escrever “OLHA O FUNIL ZAROIO”, depois ele vai simular uma digitação, depois ele espera 3 segundos e manda uma mensagem para o cliente dizendo “Eu sou o Funil Zaroio”, então ele espera 1 segundo e manda outra mensagem dizendo “Esta é a segunda mensagem do funil zaroio” para então esperar mais 3 segundos e enviar os botões com as respostas “Zaroio 1”, “Zaroio 2” e “Zaroio 3”.

Cada uma destas respostas você pode usar como gatilho de um novo bloco e desta forma você vai criando as

interações mais elaboradas. Vamos ver como isso funciona:

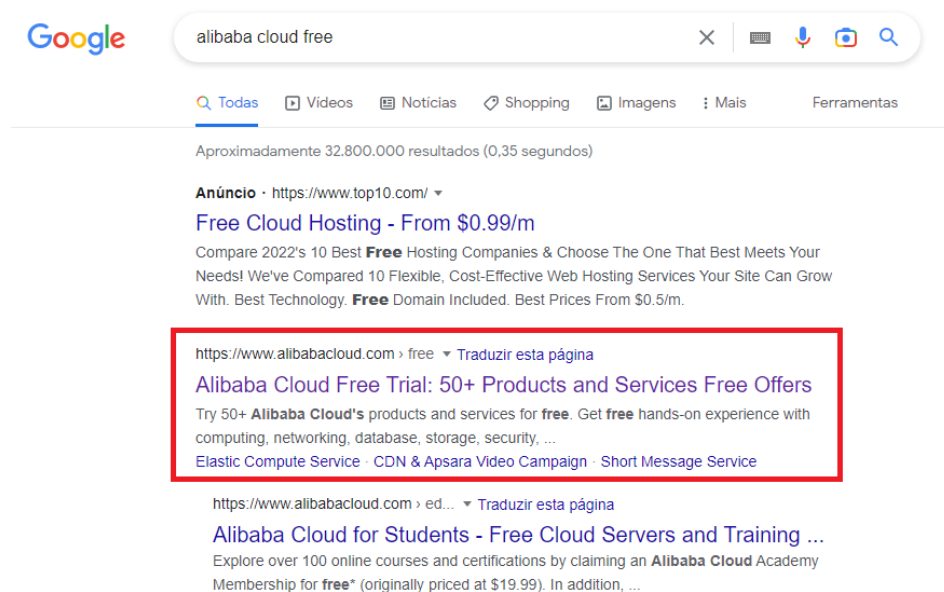
```
if (msg.body === 'Zaroio 1') {
  const chat = await msg.getChat();
  chat.sendStateTyping(); // Simulando Digitação
  await delay(3000); //Delay de 3000 milisegundos mais conhecido
  como 3 segundos
  msg.reply('Escolheu o Zaroio 1'); //Primeira mensagem de texto
  await delay(1000); //delay de 1 segundo
  client.sendMessage(msg.from, 'Olha o audio do zaroio 1');
  await delay(3000); //delay de 3 segundos
  const audio1 = MessageMedia.fromFilePath('./audio_zaroio.ogg');
  // Arquivo de audio em ogg gravado
  client.sendMessage(msg.from, audio1, {sendAudioAsVoice: true});
  // enviando o audio1
  await delay(3000); //Delay de 3 segundos
  client.sendMessage(msg.from, 'Olha a imagem do zaroio 1');
  await delay(3000); //Delay de 3 segundos
  const img1 = MessageMedia.fromFilePath('./imagem_zaroio.png'); //
  arquivo em imagem
  client.sendMessage(msg.from, img1, {caption: 'Olha que legal'});
  //Enviando a imagem
}
```

Note como o novo bloco é ligado quando respondem com “Zaroio 1” e no caso deste bloco, ele manda algumas mensagens, um áudio como se fosse gravado na hora e depois uma imagem.

- Finalizamos nosso módulo de customização e agora você está devidamente treinado para criar e explorar os seus próprios funis de atendimento automáticos. Use esta tecnologia com responsabilidade.

# Modulo Extra - Como aderir e configurar uma máquina virtual e trabalhar com seu projeto na nuvem da Alibaba

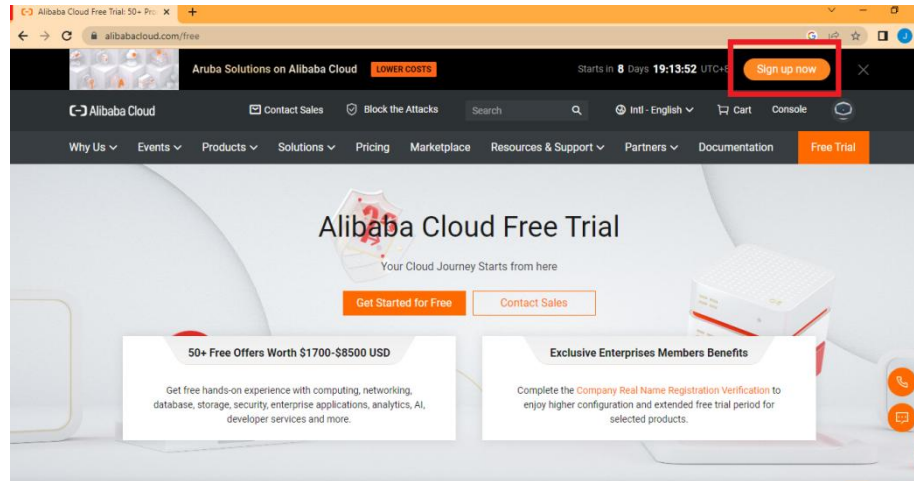
- Vamos trabalhar com a máquina virtual da Alibaba que nos oferece 12 meses de gratuidade (conferido no dia 22/11/2022). Pode ser que novos participantes apareçam nos oferecendo gratuidade e condições ainda melhores, por isso é sempre bom nos mantermos antenados no mundo da computação em nuvem.



- Vamos no Google e procuramos por “alibaba cloud free” e clicamos no link destacado em vermelho acima. Também é possível acessar

diretamente pelo endereço: <https://www.alibabacloud.com/free>

- Clique em “Sign up now” como destacado em vermelho abaixo:



- Clique em “Free Account” que te levará ao início de criação da sua conta gratuita no serviço de cloud da Alibaba como indicado na imagem abaixo:



- Aqui você fará o cadastro da forma tradicional selecionando a opção “Individual Account” ou pode simplificar ainda mais usando sua conta do gmail, bastando clicar em “Sign Up With Google” como destacado abaixo:

**Account Benefits**

**Free Trial**  
Get free hands-on experience with 50+ products. Now up to 12 Months for Elastic Compute Service!

**Premium Support Services**  
1-on-1 pre-sale consultation, 24/7 after-sales technical support with 6 free tickets per quarter

**Sign up to Alibaba Cloud**

Please select your account type \*

**Business Account**  
For purchasing services required by businesses. Enjoy premium support services and exclusive offers.

**Individual Account** ☒  
For purchasing services required by individuals or for personal use.

Next

Sign up with Google

Already a member? [Sign in](#)

- Faça todos os processos de autenticação que eles pedirem com o seu email, celular e cartão de crédito. O numero do cartão ficará linkado a sua conta caso você demande por algum serviço pago e também ao termino do período de gratuidade de 1 ano. Aqui eu recomendo que faça verificações periódicas na área de cobrança, pois é bom evitar alguma mudança repentina nas regras da empresa de cobrança. Também não esqueça que a gratuidade será validada apenas por 1 ano nesse pacote que vamos escolher, você deverá cancelar o serviço antes da cobrança começar caso assim deseje. Existem outros serviços de Cloud que oferecem promoções similares, Amazon, Microsoft, Google, todos eles possuem seus planos promocionais de gratuidade. Você pode hospedar o seu robô nesses serviços caso prefira.



### Account Benefits

**Free Trial**  
Get free hands-on experience with 50+ products. Now up to 12 Months for Elastic Compute Service!

**Premium Support Services**  
1-on-1 pre-sale consultation, 24/7 after-sales technical support with 6 free tickets per quarter

Use your Google account ( - ) to create an Alibaba Cloud account for a quick sign in.

To ensure security, specify values for the Country/Region and Password parameters. If you want to specify another email address for the Alibaba Cloud account that you want to create, [click here](#).

Select Country/Region \*

Brazil

Country/region will be used for currency and tax purposes and cannot be changed after registration.

Password \*

Confirm Password \*

☐ I hereby agree to the Alibaba Cloud International Website [Membership Agreement](#), [Privacy Policy](#), [Product Terms](#) and [Terms of Use](#), under which I am contracting with Alibaba Cloud (Singapore) Private Limited.

Sign Up

Deixe marcada a opção “Brazil” e coloque uma senha potente, confirme a senha e clique em “Sign Up”

### Basic information

Your Email: e - .

Please select your account type

**Enterprise Account**  
For purchasing services required by businesses. Enjoy premium support services and exclusive offers.

**Individual Account**  
For purchasing services required by individuals or for personal use.

[View the benefits for enterprise and individual account](#)

### Billing Information

The following information will be used for issuing invoices, please fill in with caution.

First Name \*

Last Name \*

Address line 1 \*

Street, P.O. Box, company name

Conta Individual

Escolha a opção “Individual Account” e preencha as informações conforme solicitado. Na imagem abaixo eu traduzi cada um dos campos:

**Billing Information**  
The following information will be used for issuing invoices, please fill in with caution.

First Name \*  
**Primeiro nome**

Last Name \*  
**Sobrenome**

Address line 1 \*  
**Endereço** any name

Address line 2  
Apartment, suite, unit, building, floor

City \*  
**Cidade**

State/Province \*  
**Estado**

Country/Region \*  
Brazil

Identity verification by phone \*  
+55 **Seu celular** **Verify**

Identity verification by email \*  
**Seu email** **Verify**

☐ Alibaba Cloud may not use my email to send me the latest news and deals.  
☐ Alibaba Cloud may not call me to discuss deals and offers.

**Submit**


**Verifique o código no seu celular e email**


**Após pronto, clique aqui para enviar**

A próxima tela vai pedir para escolher o meio de pagamento para aqueles que continuarão com o serviço após o período de 12 meses:

✓ Create Your Account — ✓ Basic Information — 3 Payment Information

**Add a payment method**

Credit or Debit Card   
Recommended to all customers.  
Supports JCB, VISA, MasterCard, and American Express credit or debit cards. **Add**

PayPal   
Associate your Alibaba Cloud account with your PayPal account. **Bind**

You will not be charged by Alibaba Cloud until you use cloud resources. You can manage payment methods in the Payment Methods module of the Billing Management console.

**Opção de cartão de crédito**

Na tela seguinte, a Alibaba vai enfim pedir os dados do cartão.

Traduzi os campos abaixo:

Credit or Debit Card    

Recommended to all customers.  
Supports JCB, VISA, MasterCard, and American Express credit or debit cards.

1. A bank card can only be added to one Alibaba Cloud account at a time.  
2. Prepaid cards, virtual cards, and gift cards are not supported.  
3. Please ensure that your card is activated for online payment.

Card Number \*  
**Numero do cartão**

Expiration Date \*  CVV \*   
**Mes** **Ano** **Código de verificação**

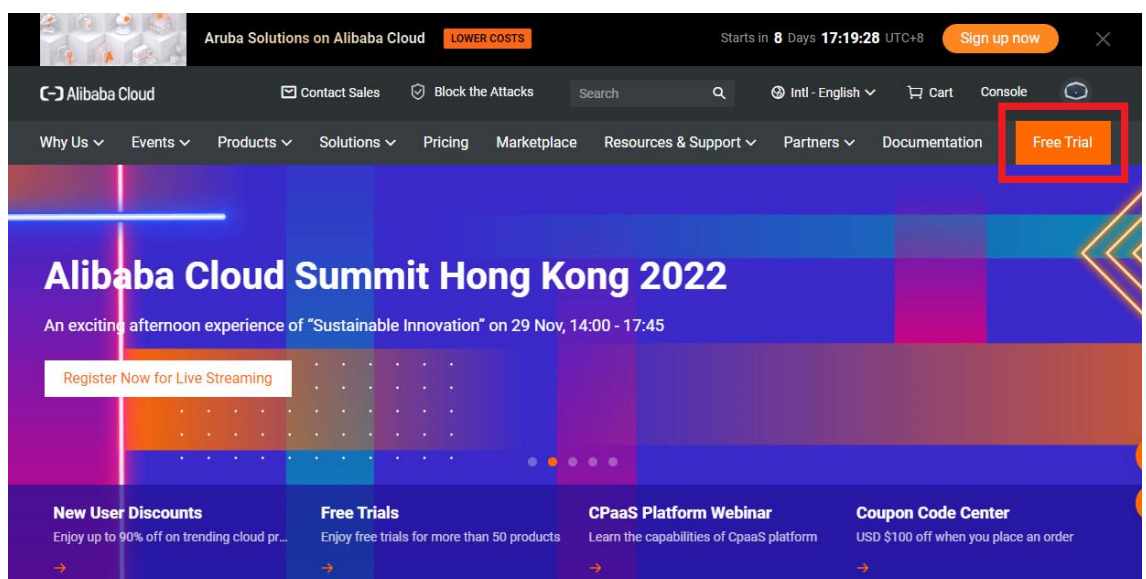
First Name \*  
**Primeiro nome**

Middle Name  
**Nome do meio**

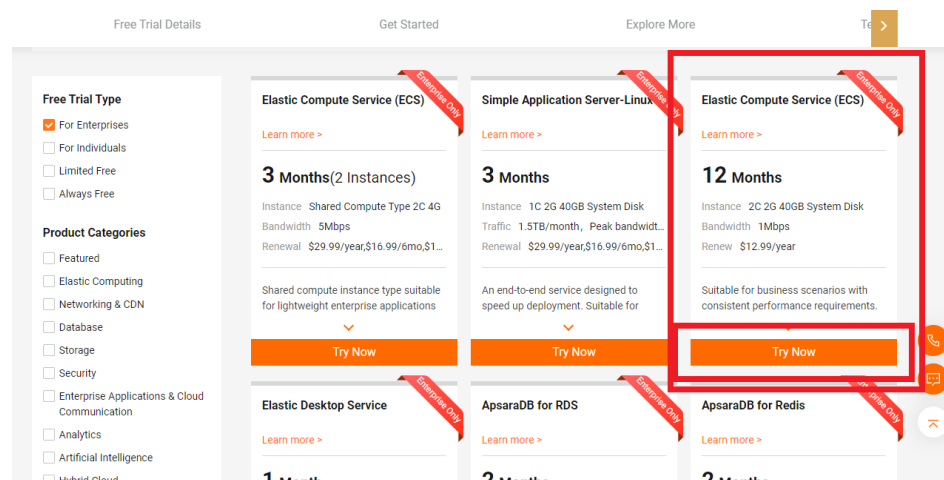
Last Name \*  
**Sobrenome**

Company Name on Card   
**Nome do banco**

- Com nosso cadastro na Alibaba devidamente configurado, clicamos em “Free Trial” na primeira tela quando logamos, isso mostrará todos os serviços gratuitos deles:



- Rolamos a tela um pouco para baixo e as opções gratuitas aparecem, vamos escolher a opção “Elastic Compute Server” com 12 meses de gratuidade clicando em “Try Now”. Ela nos libera apenas uma máquina virtual, mas o tempo de gratuidade é maior então compensa. Após isso eles cobram 219 dolares no ano (conferido em 22/11/2022)



- Com o plano selecionado. Nós escolhemos o lugar onde nosso servidor será alocado e o sistema operacional que estará dentro do servidor. No exemplo nós alocamos no Vale do Silício e estamos usando o **Windows Server 2012 em Inglês**. É importante que os parâmetros do servidor estejam iguais aos da imagem abaixo, com um Windows Server selecionado na língua Inglesa. **Então se certifique bem!**

Elastic Compute Service Free Trial (Individual) ×

Region US (Silicon Valley)

Intranets between instances in different regions do not communicate with each other; choosing a region close to your customers can reduce network latency and increase your customers' access

Available Zone Silicon Valley Zone B

Instance

1CPU 1GiB

ecs.t5-1c1m1.small

Selected Instance Type: ecs.t5-1c1m1.small (1 vCPU 1 GiB, Burstable Type t5)

OS Windows Server / Windows Server 2012 R2 Datacenter 64-bit (English)

System Disk Standard SSD 40 GiB

Total Configuration Cost **\$0.00** free for the first year Saved \$218.64

**Buy Now**

Note que o pagamento está em 0.00 dólares, mas apenas no primeiro ano de uso. Recomendo que deixe algum lembrete em seu celular para garantir o cancelamento do serviço antes da expiração da gratuidade. As outras empresas de nuvem oferecem planos de gratuidade semelhantes e ao longo do projeto eu vou experimentando outros serviços de nuvem e testando alguns truques. **Todos eles reportarei para vocês com atualizações neste guia.**

- É importante lembrar que o processo de colocar seu robô na nuvem é **totalmente opcional**, ele não é necessário para colocar o seu projeto para funcionar. Eu uso pois meus notebooks são dedicados para outras tarefas e gosto de ter a segurança de um sistema funcionando 24 horas por dia mesmo com meu computador desligado
- Como na imagem acima clicamos em “Buy Now” e aparecerá a tela de confirmação:

Elastic Compute Service Free Trial (Individual)

Unpaid Orders

Order ID	Product	Pre-tax Amount	Tax	Total Amount
	Elastic Compute Service	\$0.00	\$0.00	\$0.00

Payment Method

Add Payment Method

VISA

Total Amount: \$0.00

Purchase

Clique em “Purchase” para confirmar a compra por **zero** dólares e aparecerá a tela de confirmação e você clicará em “Console”.

Alibaba Cloud

Intl - English

Cart

Console

Pay

Confirm Order

Pay

Complete

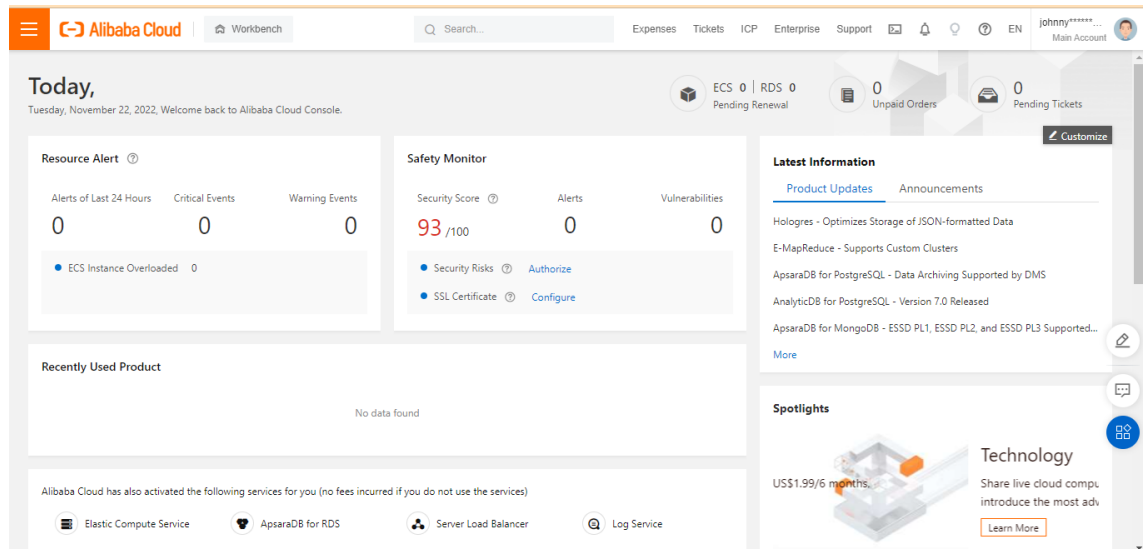
Congratulations, purchase successfully

The service you ordered is being activated. It may take 1 to 5 minutes to complete. Please wait a while.

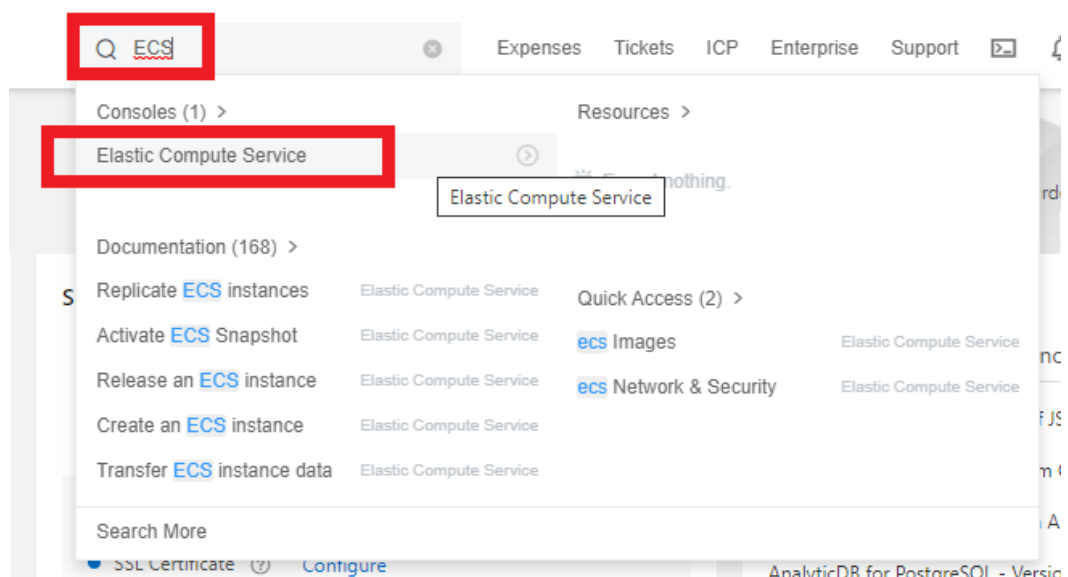
Console

Purchase History

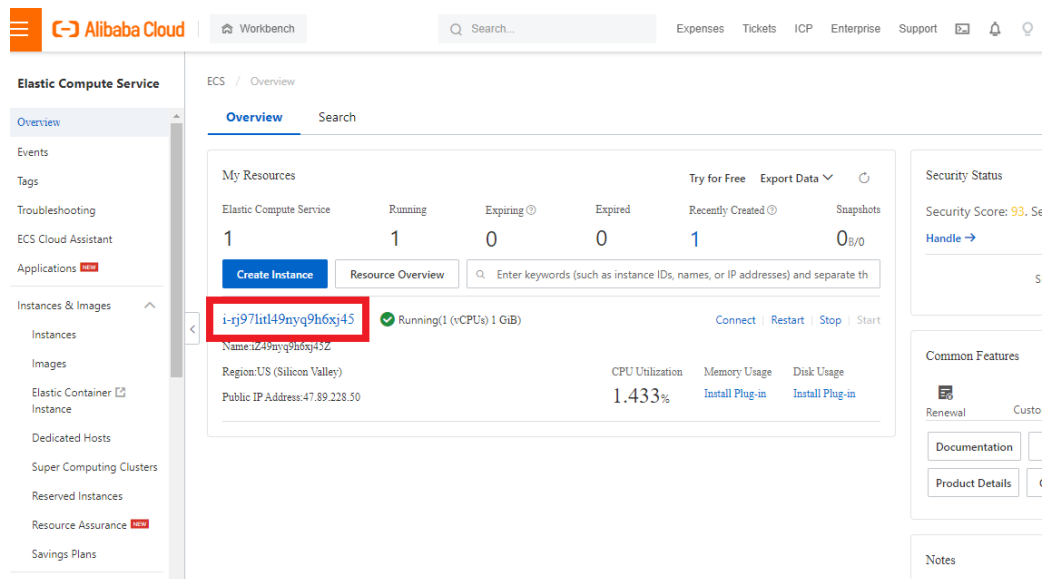
- Agora que adquirimos o serviço em nuvem gratuito, vamos deixar tudo pronto para o nosso primeiro acesso da máquina virtual. A tela de console será como essa:



- Pelo campo de busca vamos acessar o serviço de máquina virtual que acabamos de adquirir. Ele costumam levar de 10 a 20 minutos para deixar tudo configurado. Após estes minutos de espera nós digitamos “ECS” no campinho de busca:



- Clicamos sobre o nome da instancia, que neste momento está com um nome padrão todo esquisito, para abrir o seu painel. Instância é só um jeito chique de chamar o computador virtual que a Alibaba criou para nós. Na imagem abaixo você pode conferir:





- Agora que acessamos o painel da instância, vamos configurar a senha para o nosso uso dessa maquina virtual. Clique em “Reset Instance Password”:



[Instance Details](#) | [Monitoring](#) | [Security Groups](#) | [Cloud Disk](#) | [Snapshot-consistent Groups](#) | [Snapshot](#) | [E](#)

Basic Information

Start | Restart | Stop | [Configure Security Group Rule](#) | [Reset Instance Password](#)

  Running

Instance ID  
i2n-1234567890

[Connect](#)

Region  
US (Silicon Valley)

Resource Group:  
-

Zone  
Silicon Valley Zone B

Public IP  
47.100.123.456

[Convert to EIP](#)

Hostname  
i2n-1234567890

[Modify Hostname](#)

Security Group  
sg-1234567890

[Add to Security Group](#)

Created At  
Nov 22, 2022, 21:19:00 GMT-3

Description  
-

[Modify Instance Description](#)

Expires At  
Expires Nov 23, 2023, 12:59:59 GMT-3

[Renew](#)

CPU and Memory

Cloud Disk

Abrirá uma tela onde você colocará uma senha **da instância** e clicará em “Save Password”:

Reset Instance Password ?

X

If an instance is bound with a key pair, you can reset the password and use the new password to log on to the instance.

1 Reset Instance Password

2 Restart Instance

The operation will be performed on the selected 1 Instances . Are you sure you want to proceed?

ID: i2n-1234567890

Login Username: administrator

Login Username:

administrator

\* Logon Password:

.....

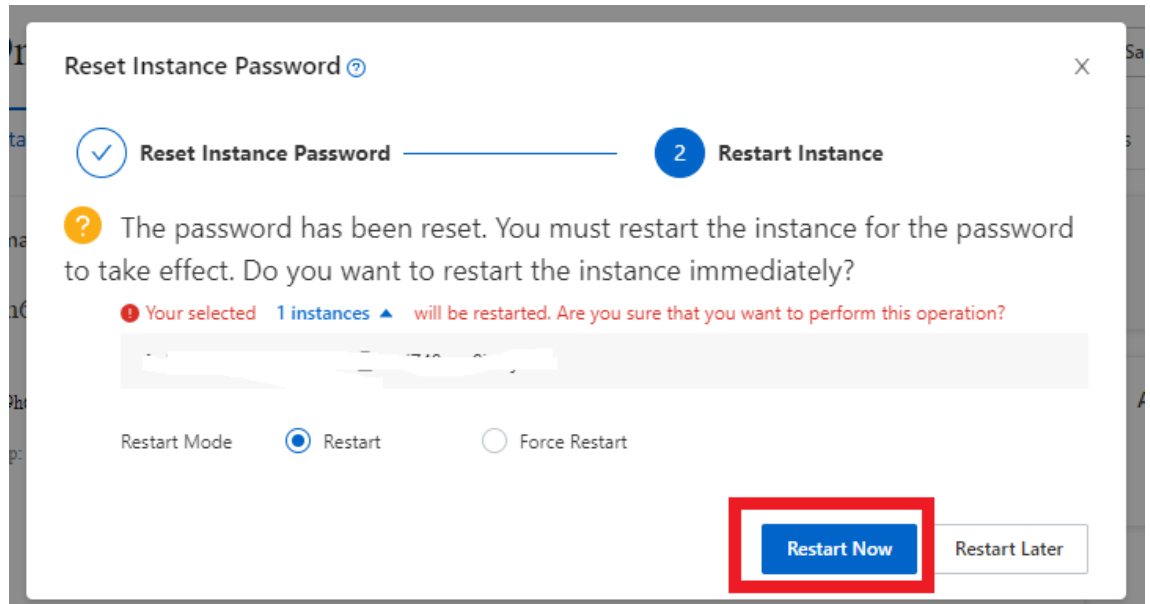
\* Confirm Password:

.....

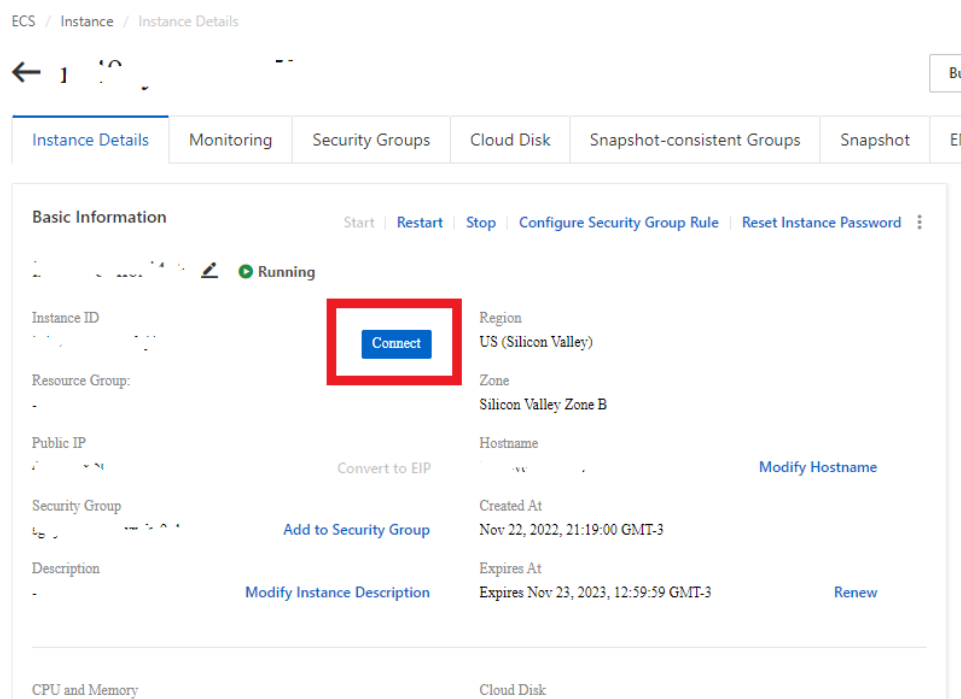
[Save Password](#)

Cancel

- Confirmamos o reset da senha com um restart da maquina virtual, clique em “Restart Now” e esperamos alguns minutos até o sistema reiniciar:



- Com a confirmação do sistema reiniciado, clicamos em “Connect” no painel da instância:



E clicamos em “Download as RDP File”:

Connection and Command ⓘ

**Workbench Connection**

ECS instances can be remotely managed by using web pages. This feature allows you to copy and paste text, and supports simultaneous logons from users on different operating systems to a single instance.

[Connect](#)

Session Manager allows you to log on to and gain remote control on an instance without a password and without security group rules configured for the instance to allow inbound access over required ports, regardless of whether the instance has a public IP address. [Learn More](#)

This feature is not supported in the current region.

**VNC Connection ⓘ**

VNC is a graphical desktop sharing tool that uses the Remote Frame Buffer (RFB) protocol to remotely access and control ECS instances. You can use VNC to connect to ECS instances without the need to configure network rules. However, VNC connections may have poor performance. We recommend that you use VNC only as a temporary connection solution.

[Connect](#)

**Send Remote Commands (Cloud Assistant) Recommended**

Cloud Assistant allows you to send remote commands to an instance and run them to perform operations such as viewing disk capacity, installing software, and starting or stopping services without connecting to the instance. To send remote commands, you must use the task execution feature provided by Cloud Assistant. Click [here](#) to install or activate the Cloud Assistant client on your instance.

[Send Remote Commands](#)

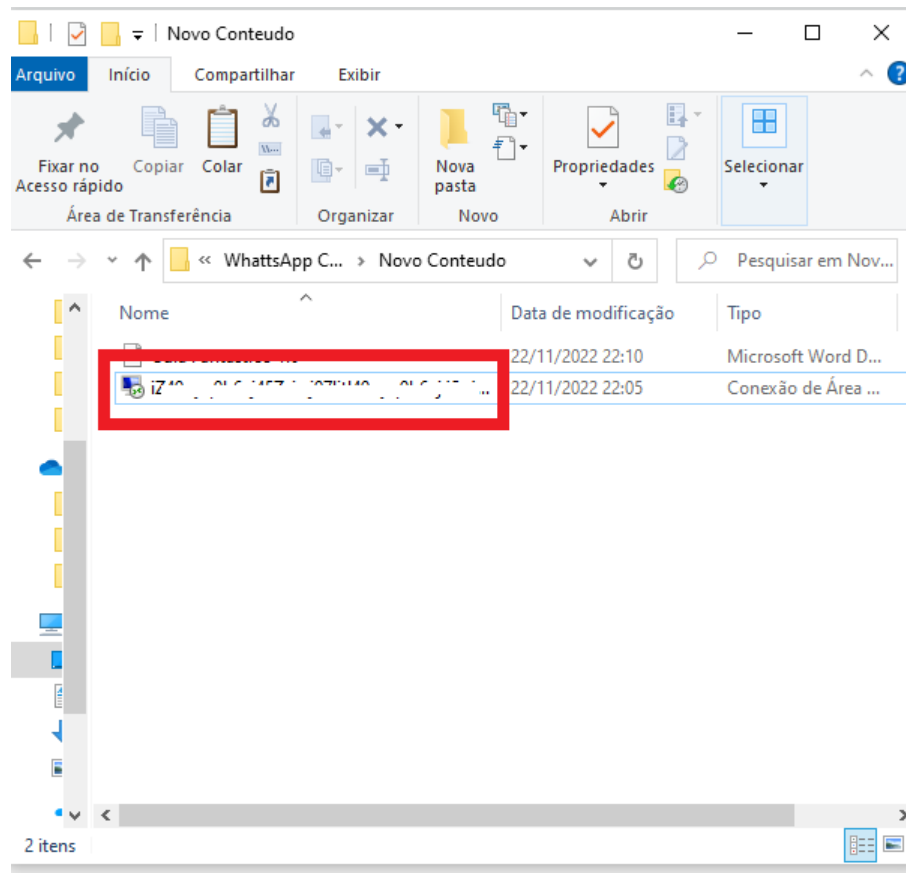
[Download as RDP File](#)

[Troubleshoot](#) [Cancel](#)

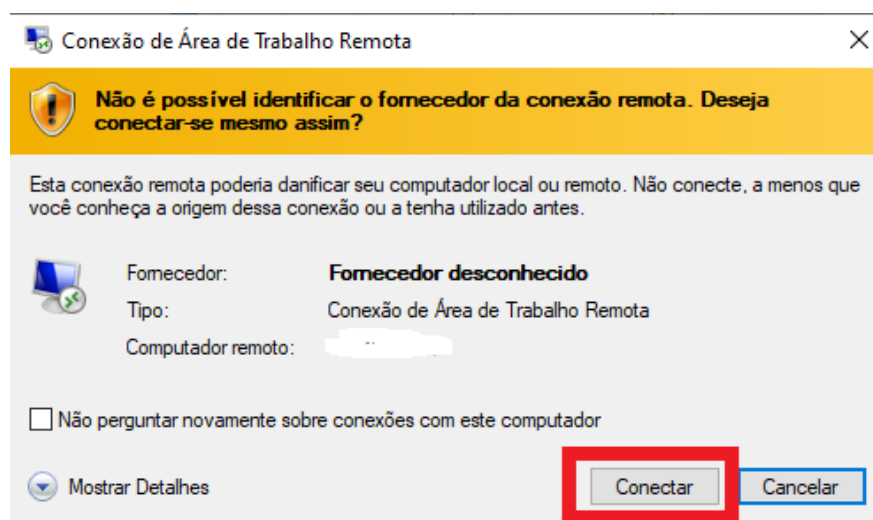
O arquivo RDP é um caminho fácil que vai nos apontar para a máquina virtual com rapidez, ao clicar nesse link, o arquivo será baixado e toda vez que precisarmos acessar a máquina virtual, nós usaremos este arquivo. Por isso é importante que ele esteja organizado numa pasta fácil de ser encontrada.



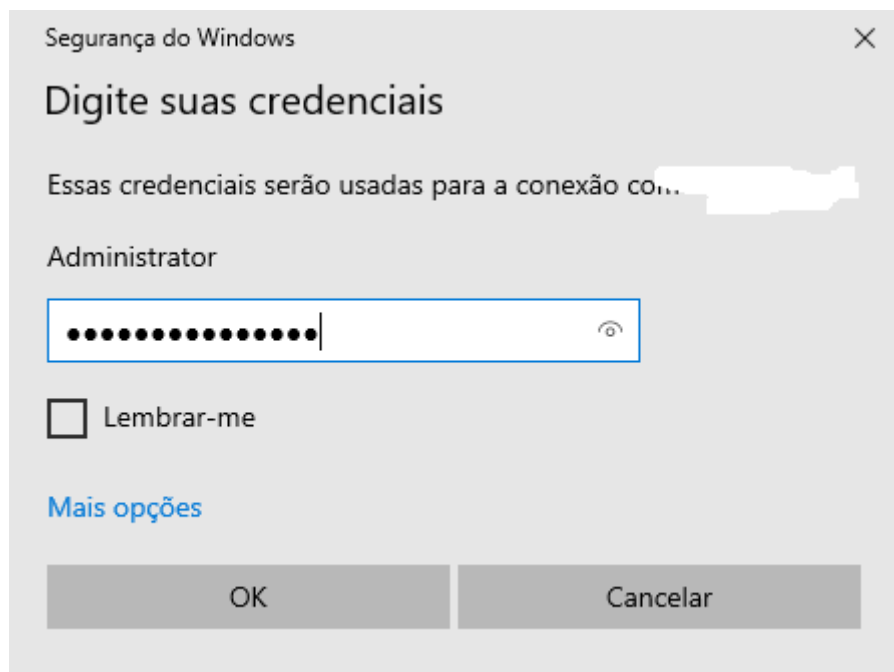
Assim:



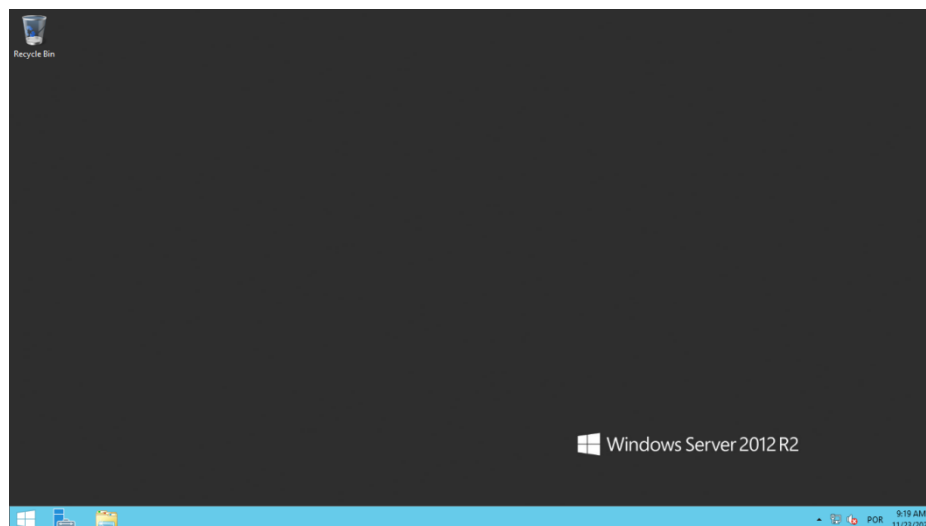
- Com dois cliques no arquivo e é aberto o portal para o mundo da nossa máquina virtual, aceitamos clicando em “Conectar”:



Note que a senha é a mesma que você configurou logo antes quando resetamos a instancia:



Clique em “OK” e confirme, logo em seguida, todas as confirmações de acesso e protocolos que aparecer. Ao final você deverá ter uma tela como essa:



Nosso Windows Server 2012 está configuradinho na máquina virtual. E está tudo pronto para prepararmos nossa estrutura de instalação no mesmo formato de quando configuramos na máquina local.

# Anexo Código Base – Copie e Cole o Código Base

```
// Invocamos o leitor de qr code

const qrcode = require('qrcode-terminal');
const { Client, Buttons, List, MessageMedia } = require('whatsapp-web.js'); // Mudança Buttons
const client = new Client();
// então habilitamos o usuário a acessar o serviço de leitura do qr code
client.on('qr', qr => {
  qrcode.generate(qr, {small: true});
});
// após isso ele diz que foi tudo certo
client.on('ready', () => {
  console.log('Tudo certo! WhatsApp conectado.');
```

});

// E inicializa tudo para fazer a nossa magia =)

client.initialize();

// Funil Base Projeto

const delay = ms => new Promise(res => setTimeout(res, ms)); // Função que usamos para criar o delay entre uma ação e outra

client.on('message', async msg => {

if (msg.body === 'ATIVAR FUNIL BASICO') {

const chat = await msg.getChat();

chat.sendStateTyping(); // Simulando Digitação

await delay(3000); // Delay de 3000 milisegundos mais conhecido como 3 segundos

msg.reply('Olá! Seja muito bem vindo. Você entrou no Funil Basico do treinamento Chatbot projetado pelo Johnny'); // Primeira mensagem de texto

await delay(1000); // delay de 1 segundo

client.sendMessage(msg.from, 'Você vai ter contato com as funcionalidades básicas do nosso projeto e poderá ver o quanto é fácil criar seus próprios funis personalizados ao seu negócio.');

await delay(3000); // delay de 3 segundos

client.sendMessage(msg.from, 'Agora vamos testar os botões. 📄');

client.sendMessage(msg.from, new Buttons('Olha que bacana', [{id: 'customId', body: 'Bacana demais!!'}, {body: 'Eu concordo, mto mesmo..'}], 'Vamos lá...', 'Escolha abaixo 📄'));

```

    }

    if (msg.body !== null && msg.body === 'Bacana demais!!') {
        const chat = await msg.getChat();
        chat.sendStateTyping(); //Simulando digitação
        await delay(3000); //Delay de 3 segundos
        client.sendMessage(msg.from, 'Você escolheu a opção Bacana
        demais. Isso é muito bom, pois na prática você vai se comunicar com seus
        clientes exatamente desta maneira.');
```

await delay(3000); //Delay de 3 segundos

client.sendMessage(msg.from, 'Agora eu vou te mandar um audio
 gravado mas enviado como se fosse fresquinho!!');

chat.sendStateRecording(); //Simulando audio gravando

await delay(5000); //Delay de 5 segundos

const form1 = MessageMedia.fromFilePath('./audio\_base.ogg'); //
 Arquivo de audio em ogg gravado

client.sendMessage(msg.from, form1, {sendAudioAsVoice: true});

// enviando o audio1

await delay(4000); //Delay de 4 segundos

client.sendMessage(msg.from, 'Agora quero te mandar uma imagem');

await delay(3000); //Delay de 3 segundos

const img1 = MessageMedia.fromFilePath('./imagem\_base.png'); //
 arquivo em imagem

client.sendMessage(msg.from, img1, {caption: 'Olha que legal'});

//Enviando a imagem

await delay(3000); //Delay de 3 segundos

client.sendMessage(msg.from, 'Prontinho! Agora use a sua
 criatividade para criar sequencias de respostas com audios, botões e
 imagens. O céu é o limite');

}

```

    if (msg.body !== null && msg.body === 'Eu concordo, mto mesmo..') {
        const chat = await msg.getChat();
        chat.sendStateTyping(); //Simulando digitação
        await delay(3000); //Delay de 3 segundos
        client.sendMessage(msg.from, 'Você escolheu a opção Eu concordo,
        mto mesmo.. Isso é muito bom, pois na prática você vai se comunicar com
        seus clientes exatamente desta maneira.');
```

await delay(3000); //Delay de 3 segundos

client.sendMessage(msg.from, 'Agora eu vou te mandar um audio
 gravado mas enviado como se fosse fresquinho!!');

chat.sendStateRecording(); //Simulando audio gravando

await delay(5000); //Delay de 5 segundos

const form1 = MessageMedia.fromFilePath('./audio\_base.ogg'); //
 Arquivo de audio em ogg gravado

client.sendMessage(msg.from, form1, {sendAudioAsVoice: true});

// enviando o audio1

await delay(4000); //Delay de 4 segundos

client.sendMessage(msg.from, 'Agora quero te mandar uma imagem');

```
        await delay(3000); //Delay de 3 segundos
        const img1 = MessageMedia.fromFilePath('./imagem_base.png'); //
arquivo em imagem
        await delay(3000); //Delay de 3 segundos
        client.sendMessage(msg.from, 'Prontinho! Agora use a sua
criatividade para criar sequencias de respostas com audios, botões e
imagens. O céu é o limite');
    }
});
```