1- criando uma notificação de ordem de compra se ela foi feita com sucesso: módulos Proposta —> Módulo customizado Purchase Order. joão

```
/*
Function made by João Vitor CRM7
2022 May, 11.
*/
//Global Variables//
quote = zoho.crm.getRecordById("Propostas",quoteID);
fabricanteID = quote.get("Fabricante").get("id");
info fabricanteID;
fabricante = zoho.crm.getRecordById("Vendors",fabricanteID);
//
if(quote.get("status") == "failure")
{
	return "[Erro] - A Proposta se encontra em processo de aprovação, aguarde até que a mesma seja aprovada para prosseguir com a conversão.";
}
else
{
	info quote;
	quoteStatus = quote.get("Status").toLowerCase();
	quoteType = quote.get("Prenchimento_tabela");
	//Tasks//
	if(quoteStatus != "fechado")
	{
		return "[Erro] - Essa Proposta não se encontra no status Fechado.";
	}
	else if(quoteType == "serviços")
	{
		return "[Erro] - O preenchimento dessa Proposta é de Serviços.";
	}
	else
	{
		//Field Mapping//
		purchaseOrder = Map();
		//Registro Section//
		purchaseOrder.put("Owner",quote.get("Owner"));
		purchaseOrder.put("Status","Rascunho");
		purchaseOrder.put("Contato",quote.get("field"));
		//Informações da Purchase Order PO Section//
		purchaseOrder.put("Name",quote.get("Name"));
```

```
purchaseOrder.put("Validade_da_Purchase_Order",quote.get("Validade_da_Proposta"));
                    purchaseOrder.put("Neg_cios",quote.get("Contato"));
                    purchaseOrder.put("Cliente",quote.get("Cliente"));
                    purchaseOrder.put("Purchase_Orders_Enviada","Não");
                    purchaseOrder.put("Proposta",quote.get("id"));
                    //Itens Cotados de Produto Section//
                    subformData = quote.get("Itechns_Cotados_de_Produtos");
                    purchaseOrder.put("Itens_Cotados_Produto",subformData);
                    //Valores Totais Produtos Section//
                    purchaseOrder.put("Receita_Produto_R",quote.get("Receita_Produtos"));
                    purchaseOrder.put("Custo_Produtos_R",quote.get("Custo_Produtos"));
                    purchaseOrder.put("Imposto_Produtos_R",quote.get("Imposto_Produtos"));

purchaseOrder.put("Resultado_Produto_R",quote.get("Resultado_Produtos"));

purchaseOrder.put("Termos_e_Condi_es_Produto",quote.get("Termos_e_Condi_es_Produto
"));
                    purchaseOrder.put("Pagamento",quote.get("Pagamento"));
                    purchaseOrder.put("Faturamento",quote.get("Faturamento"));
                    //Fabricante Section//
                    purchaseOrder.put("Empresa_de_Compra",fabricanteID);
                    purchaseOrder.put("Street_Fabricante",fabricante.get("Street"));
                    purchaseOrder.put("City_Fabricante",fabricante.get("City"));
                    purchaseOrder.put("Estado_do_Fabricante",fabricante.get("State"));
                    purchaseOrder.put("Country_Fabricante",fabricante.get("Country"));
                    purchaseOrder.put("Net_Terms",fabricante.get("Net_Terms"));
                    //
                    purchaseOrderCreation =
zoho.crm.createRecord("Purchase_Orders_PO",purchaseOrder);
                    //
                    return "A Purchase Order foi criada com sucesso, você pode encontra-la
atrelada nessa Proposta.";
          }
}
return 0;
```

1.2 Gustavo

```
//Definindo variaveis Globais
proposta = zoho.crm.getRecordById("Propostas", idProposta);
// info proposta;

if ( proposta.get("Status") != "Fechado")
{
          return "A proposta não está Fechada.";
```

```
}else{

        purchaseOrder = Map();

        // sessão registro Purchase Order PO Owner Owner
        purchaseOrder.put("Owner",proposta.get("Owner");

        // sessão informações da Purchase Order PO
        purchaseOrder.put("Name",proposta.get("Name"));
        purchaseOrder.put("Neg_cios",proposta.get("Neg_cios"));
        purchaseOrder.put("Email",proposta.get("Email      "));
        purchaseOrder.put("Cliente",proposta.get("Cliente"));
        purchaseOrde.put("Proposta",proposta.get("Proposta"));

purchaseOrder.put("Validade_da_Purchase_Order",proposta.get("Validade_da_Purchase_O
rder"));
        purchaseOrder.put("Purchase_Orders_Enviada",
proposta.get("Purchase_Orders_Enviada"));

        //Itens Cotados de Produto
        itensCotado = proposta.get("Itens_Cotados_Produto");
        purchaseOrder.put("Itens_Cotados_Produto",itensCotado);

        //Fabricante
        //Endereço Empresa de Compra
        //Distribuidor
        //Revenda

        purchaseOrderCreation =
zoho.crm.createRecord("Purchase_Orders_PO",purchaseOrder);

        return "A Purchase Order foi criada";
}

return 0;
```

links do projeto que estou trabalhando

https://crmsandbox.zoho.com/crm/sandboxv2k/tab/CustomModule12/create?layoutId=5029274000000397267

2-Implementando um botão de verificação do CNPJ, consumindo uma API, Criar os campos necessários para receber esses dados e depois fazer a tratativa desses dados.

```
cliente = zoho.crm.getRecordById("Leads",idClientes);
// info  cliente;
cnpj = cliente.get("Cnpj_Api");
info cnpj;

cnpjFormatado = cnpj.replaceAll("[a-zA-z!@#$%^&*.-/////-]","");

header_data  = Map();
header_data.put("x-api-key","cd6de8c5-bb55-4fdd-b0cc-fbd4c616d8da");

response = invokeUrl
[
	url:  "https://www.receitaws.com.br/v1/cnpj/"+cnpjFormatado
	type: GET

];

info response;
return response;
```

2.1 Versão um pouco mais completa Final!!!!!

```
cliente = zoho.crm.getRecordById("Leads",idClientes);
// info  cliente;
cnpj = cliente.get("Cnpj_Api");
// info cnpj;
cnpjFormatado = cnpj.replaceAll("[a-zA-z!@#$%^&*.-/////-]","");
header_data = Map();
header_data.put("x-api-key","cd6de8c5-bb55-4fdd-b0cc-fbd4c616d8da");
response = invokeurl
[
	url :"https://www.receitaws.com.br/v1/cnpj/" + cnpjFormatado
	type :GET
];
info response;
//  info response;
```

```
// return response;
//adicionar dados da API para um módulo
// dadosCnpj.put("Abertura",response.get("Abertura"));
dataAbertura = response.get("abertura");
logradouro = response.get("logradouro");

// info dataAbertura;
// info logradouro;

MapaDadosCnpj = Map();
MapaDadosCnpj.put("Cnpj",response.get("cnpj"));
MapaDadosCnpj.put("Abertura",response.get("abertura"));
MapaDadosCnpj.put("Logradouro",response.get("logradouro"));
MapaDadosCnpj.put("Numero",response.get("numero"));
MapaDadosCnpj.put("Cep",response.get("cep"));
MapaDadosCnpj.put("Municipio",response.get("municipio"));
MapaDadosCnpj.put("Bairro",response.get("bairro"));
MapaDadosCnpj.put("Uf",response.get("uf"));
MapaDadosCnpj.put("Ultima_Atualizacao",response.get("ultima_atualizacao"));
MapaDadosCnpj.put("Empresa",response.get("natureza_juridica"));
MapaDadosCnpj.put("Last_Name",response.get("fantasia"));

MapaCreation = zoho.crm.updateRecord("Leads",idClientes,MapaDadosCnpj);
info MapaCreation;

return "Os dados foram importados com sucesso!";
```

3- consumir os dados da API de CEP, mapear os campos e criar campos correspondentes:

```
contato = zoho.crm.getRecordById("Contacts",idContato);
// info contato;
cep = contato.get("Mailing_Zip");
// info cep;
cepFormatado = cep.replaceAll("[A-Za-z!@#$%¨&*-.,;/]","");
// info cepFormatado;
dadosCep = invokeurl
[
        url :"http://viacep.com.br/ws/" + cepFormatado + "/json/"
        type :GET
];
info dadosCep;
info dadosCep;
mapaCep = Map();
```

```
mapaCep.put("Mailing_City",dadosCep.get("localidade"));
mapaCep.put("Complemento",dadosCep.get("complemento"));
mapaCep.put("Bairrro",dadosCep.get("bairro"));
mapaCep.put("Gia",dadosCep.get("gia"));
mapaCep.put("Ibge",dadosCep.get("ibge"));
updateMapCep = zoho.crm.updateRecord("Contacts",idContato,mapaCep);
info updateMapCep;
return "";
```

4- Consumindo API do CEP e criando campos específicos para receber os dados:

```
5contato = zoho.crm.getRecordById("Contacts",idContato);
// info contato;
cep = contato.get("Mailing_Zip");
// info cep;
cepFormatado = cep.replaceAll("[A-Za-z!@#$%¨&*-.,;/]","");
// info cepFormatado;
dadosCep = invokeurl
[
	url :"http://viacep.com.br/ws/" + cepFormatado + "/json/"
	type :GET
];
info dadosCep;
info dadosCep;
mapaCep = Map();
mapaCep.put("Mailing_City",dadosCep.get("localidade"));
mapaCep.put("Complemento",dadosCep.get("complemento"));
mapaCep.put("Bairrro",dadosCep.get("bairro"));
mapaCep.put("Gia",dadosCep.get("gia"));
mapaCep.put("Ibge",dadosCep.get("ibge"));
updateMapCep = zoho.crm.updateRecord("Contacts",idContato,mapaCep);
info updateMapCep;
return "";
```

5-Criando um Client Script (”Script do Cliente”), na verificação de validação do CPF.

```
const campo_cpf = ZDK.Page.getField('CPF_CS').getValue();
const cpfFormatado = campo_cpf.replaceAll(/[a-zA-z!@#$%^&*.-/////-]/g, "");
```

```
console.log("esse é o valor do campo Formatado", campo_cpf);

console.log(cpfFormatado.toString().length);

const tamanho_cpf_formtado = cpfFormatado.toString().length;

if (cpfFormatado == 'null'|| cpfFormatado == '' ){
   return ZDK.Client.showAlert('Campo Vazio!');

} if ( tamanho_cpf_formtado== 11) {
    ZDK.Client.showAlert('11 Digitos CPF:' + cpfFormatado);

        num1 = cpfFormatado.substr(0,1);
        num2 = cpfFormatado.substr(1,1);
        num3 = cpfFormatado.substr(2,1);
        num4 = cpfFormatado.substr(3,1);
        num5 = cpfFormatado.substr(4,1);
        num6 = cpfFormatado.substr(5,1);
        num7 = cpfFormatado.substr(6,1);
        num8 = cpfFormatado.substr(7,1);
        num9 = cpfFormatado.substr(8,1);
        num10 = cpfFormatado.substr(9,1);
        num11 = cpfFormatado.substr(10,1);


        num1Int = parseInt(num1);
              num2Int = parseInt(num2);
              num3Int = parseInt(num3);
              num4Int = parseInt(num4);
              num5Int = parseInt(num5);
              num6Int = parseInt(num6);
              num7Int = parseInt(num7);
              num8Int = parseInt(num8);
              num9Int = parseInt(num9);
              num10Int = parseInt(num10);
              num11Int = parseInt(num11);


    calc1 = num1Int*10; calc2 = num2Int*9;
    calc3 = num3Int * 8;calc4 = num4Int * 7;
          calc5 = num5Int*6;calc6 = num6Int * 5;calc7 = num7Int * 4;calc8 = num8Int *
3;
          calc9 = num9Int*2;
```

```
        somandoDigitos01 = calc1 + calc2 + calc3 + calc4 + calc5 + calc6 + calc7 +
calc8 + calc9;

        divisao01 = (somandoDigitos01*10)/11;
        divisaoResto = (somandoDigitos01*10)%11;

   console.log("Somatoria: " + somandoDigitos01);
   console.log("Divisao: " + divisao01);
   console.log("Divisao Resto: " + divisaoResto);

   if(divisaoResto == 10 || divisaoResto == 11 || divisaoResto == num10Int)
        {
                console.log("uma das validaçoes foi aceita");
                console.log("Calculando o segundo Digito!");

                calc1 = num1Int * 11;calc2 = num2Int * 10;calc3 = num3Int * 9;calc4 =
num4Int * 8;calc5 = num5Int * 7;
                calc6 = num6Int * 6;calc7 = num7Int * 5;calc8 = num8Int * 4;calc9 =
num9Int * 3;calc10 = num10Int * 2;
                somandoDigitos2 = calc1 + calc2 + calc3 + calc4 + calc5 + calc6 +
calc7 + calc8 + calc9 + calc10;

                divisao2 = (somandoDigitos2 * 10) / 11;
                divisaoResto2 = (somandoDigitos2 * 10) % 11;
                console.log("Valor da divisao: "+divisao2);
                console.log("Resto da divisão: " +divisaoResto2) ;


                if(divisaoResto2 == 10 || divisaoResto2 == 11 || divisaoResto2 ==
num11Int)
                {

                        ZDK.Client.showAlert("CPF ACEITO PARABENS!!");
                }else
                {

                        ZDK.Client.showAlert("CPF INVALIDO! ");
                }

        }else{

      ZDK.Client.showAlert("CPF INVALIDO! ");
   }
```

```
}
else{
    ZDK.Client.showAlert('CPF INVALIDO! '+cpfFormatado);

}
```

5 versão 2 JavaCript

```
function verificaCpf() {

    const numeroCpfId = document.getElementById('numeroCpf').value

    const  cpfFormatado = numeroCpfId.replaceAll(/[a-zA-z!@#$%^&*.-/////-]/g, "");

    const tamanhoCpfFormatado = cpfFormatado.toString().length;

    if(cpfFormatado == "00000000000" || cpfFormatado == "11111111111" ||
cpfFormatado == "22222222222" || cpfFormatado == "33333333333" ||
cpfFormatado == "44444444444" || cpfFormatado == "55555555555" ||
cpfFormatado == "66666666666" || cpfFormatado == "77777777777" ||
cpfFormatado == "88888888888" || cpfFormatado == "99999999999")
{
          return  alert ("valores invalidos padroes XXX XXX XXX XX");
}

    if(numeroCpfId== 'null' || numeroCpfId == ""){


      return alert("Campo Vazio!"+numeroCpfId);
 }

    if (tamanhoCpfFormatado == 11){

        alert("11 Digitos:  "+cpfFormatado);

        document.getElementById("cpf_formatado").innerHTML =  cpfFormatado;
```

```
        num1 = cpfFormatado.substr(0,1);
        num2 = cpfFormatado.substr(1,1);
        num3 = cpfFormatado.substr(2,1);
        num4 = cpfFormatado.substr(3,1);
        num5 = cpfFormatado.substr(4,1);
        num6 = cpfFormatado.substr(5,1);
        num7 = cpfFormatado.substr(6,1);
        num8 = cpfFormatado.substr(7,1);
        num9 = cpfFormatado.substr(8,1);
        num10 = cpfFormatado.substr(9,1);
        num11 = cpfFormatado.substr(10,1);


        num1Int = parseInt(num1);
             num2Int = parseInt(num2);
             num3Int = parseInt(num3);
             num4Int = parseInt(num4);
             num5Int = parseInt(num5);
             num6Int = parseInt(num6);
             num7Int = parseInt(num7);
             num8Int = parseInt(num8);
             num9Int = parseInt(num9);
             num10Int = parseInt(num10);
             num11Int = parseInt(num11);


    calc1 = num1Int*10; calc2 = num2Int*9;
    calc3 = num3Int * 8;calc4 = num4Int * 7;
         calc5 = num5Int*6;calc6 = num6Int * 5;calc7 = num7Int * 4;calc8 = num8Int *
3;
         calc9 = num9Int*2;

         somandoDigitos01 = calc1 + calc2 + calc3 + calc4 + calc5 + calc6 + calc7 +
calc8 + calc9;

         divisao01 = (somandoDigitos01*10)/11;
         divisaoResto = (somandoDigitos01*10)%11;

         document.getElementById("ver1").innerHTML = somandoDigitos01;
         document.getElementById("ver2").innerHTML = divisao01;
    document.getElementById("ver3").innerHTML =  divisaoResto;

    if(divisaoResto == 10 || divisaoResto == 11 || divisaoResto == num10Int)
```

```
        {
                console.log("uma das validaçoes foi aceita");
                console.log("Calculando o segundo Digito!");

                calc1 = num1Int * 11;calc2 = num2Int * 10;calc3 = num3Int * 9;calc4 =
num4Int * 8;calc5 = num5Int * 7;
                calc6 = num6Int * 6;calc7 = num7Int * 5;calc8 = num8Int * 4;calc9 =
num9Int * 3;calc10 = num10Int * 2;
                somandoDigitos2 = calc1 + calc2 + calc3 + calc4 + calc5 + calc6 +
calc7 + calc8 + calc9 + calc10;

                divisao2 = (somandoDigitos2 * 10) / 11;
                divisaoResto2 = (somandoDigitos2 * 10) % 11;
                console.log("Valor da divisao: "+divisao2);
                console.log("Resto da divisão: " +divisaoResto2) ;


                if(divisaoResto2 == 10 || divisaoResto2 == 11 || divisaoResto2 ==
num11Int)
                {

                        console.log("CPF ACEITO PARABENS!!");
                }else
                {

                        console.log("CPF INVALIDO! ");
                }

        }
}
   else{
      alert("Caracteres inválidos!");
      console.log(cpfFormatado);
   }
}
```

6-Criar um Script em "Script de Cliente" que valide e mascare o CNPJ que foi digitado na Hora da criação do Registro.

- [ ] Mascara
- [ ] Validação do CNPJ
- [ ] Retornos de CNPJ invalidados

- [ ] Retorno Sucesso

Código Client Script

```
const campoCnpj = ZDK.Page.getField('CNPJ_CS').getValue();

const cnpjFormatado = campoCnpj.replaceAll(/[a-zA-z!@#$%^&*.-/////-]/g, "");

const tamanhoCnpjFormatado = cnpjFormatado.toString().length;




if (cnpjFormatado == 'null' || cnpjFormatado == "") {

    return ZDK.Client.showAlert("CNPj inválido" + cnpjFormatado +" digitos"
+tamanhoCnpjFormatado);

}



if(cnpjFormatado == "00000000000000" || cnpjFormatado == "11111111111111" ||
cnpjFormatado == "22222222222222" || cnpjFormatado == "33333333333333" ||
cnpjFormatado == "44444444444444" || cnpjFormatado == "55555555555555" ||
cnpjFormatado == "66666666666666" || cnpjFormatado == "77777777777777" ||
cnpjFormatado == "88888888888888" || cnpjFormatado == "99999999999999")



{

  return ZDK.Client.showAlert("Valor inválido de Cnpj!"+cnpjFormatado +" digitos"
+tamanhoCnpjFormatado);

}



if (tamanhoCnpjFormatado == 14) {



        num1 = cnpjFormatado.substr(0,1);

        num2 = cnpjFormatado.substr(1,1);

        num3 = cnpjFormatado.substr(2,1);

        num4 = cnpjFormatado.substr(3,1);
```

```
        num5 = cnpjFormatado.substr(4,1);

        num6 = cnpjFormatado.substr(5,1);

        num7 = cnpjFormatado.substr(6,1);

        num8 = cnpjFormatado.substr(7,1);

        num9 = cnpjFormatado.substr(8,1);

        num10 = cnpjFormatado.substr(9,1);

        num11 = cnpjFormatado.substr(10, 1);

        num12 = cnpjFormatado.substr(11, 1);

        num13 = cnpjFormatado.substr(12, 1);

        num14 = cnpjFormatado.substr(13, 1);



      num1Int = parseInt(num1);

      num2Int = parseInt(num2);

      num3Int = parseInt(num3);

      num4Int = parseInt(num4);

      num5Int = parseInt(num5);

      num6Int = parseInt(num6);

      num7Int = parseInt(num7);

      num8Int = parseInt(num8);

      num9Int = parseInt(num9);

      num10Int = parseInt(num10);

      num11Int = parseInt(num11);

         num12Int = parseInt(num12);

         num13Int = parseInt(num13);

  num14Int = parseInt(num14);



        // _____________________________Primeiro Digito
Verificador_____________________________
```

```
    calc1 = num1Int * 5;calc2 = num2Int * 4;calc3 = num3Int * 3;calc4 = num4Int *
2;calc5 = num5Int * 9;

  calc6 = num6Int * 8;calc7 = num7Int * 7;calc8 = num8Int * 6;calc9 = num9Int *
5;calc10 = num10Int * 4;

  calc11 = num11Int * 3;calc12 = num12Int * 2;



  const somandoDigitos01 = calc1 + calc2 + calc3 + calc4 + calc5 + calc6 + calc7 +
calc8 + calc9 + calc10 + calc11 + calc12;




  console.log(somandoDigitos01);

  divisao01 = somandoDigitos01/11;

  divisaoResto = somandoDigitos01%11;

  console.log("Divisão: " + divisao01);

  console.log("Resto da divisão:" + divisaoResto);

  const primeiroDigitoVerificador = 11 - divisaoResto;




  //_____________________________________Segundo Digito
Verificador_____________________________




  calc1 = num1Int * 6;

      calc2 = num2Int * 5;

      calc3 = num3Int * 4;

      calc4 = num4Int * 3;

      calc5 = num5Int * 2;

      calc6 = num6Int * 9;
```

```
calc7 = num7Int * 8;

calc8 = num8Int * 7;

calc9 = num9Int * 6;

calc10 = num10Int * 5;

calc11 = num11Int * 4;

calc12 = num12Int * 3;

calc13 = num13Int * 2;



const somandoDigitos02 =

  calc1 +

  calc2 +

  calc3 +

  calc4 +

  calc5 +

  calc6 +

  calc7 +

  calc8 +

  calc9 +

  calc10 +

  calc11 +

  calc12 +

  calc13;



console.log(somandoDigitos02);



divisao02 = somandoDigitos02 / 11;

divisaoResto02 = somandoDigitos02 % 11;
```

```
      console.log("Divisão2: " + divisao02);
      console.log("Resto da divisão2:" + divisaoResto02);
      const segundoDigitoVerificador = 11 - divisaoResto02;

   if (divisaoResto == 0 || divisaoResto == 1) {
        console.log("1° Digito verificador Válido =0");

        if (divisaoResto02 == 0 || divisaoResto02 == 1) {
          console.log("2° Digito verificador Válido =0");
          return ZDK.Client.showAlert("CNPJ Aceito");
        } else {
          return ZDK.Client.showAlert("CNPJ 2°Digito verificador invalido!");
      }
}if (divisaoResto <= 10 && divisaoResto >= 2 &&
primeiroDigitoVerificador==num13Int) {

        console.log(
          " O Primeiro digito verificador é: " + primeiroDigitoVerificador
        );

        if (divisaoResto02 <= 10 && divisaoResto02 >= 2 &&
segundoDigitoVerificador==num14Int) {

          console.log(
            " O segundo digito verificador é: " + segundoDigitoVerificador
          );
          return ZDK.Client.showAlert("CNPJ Aceito");
```

```
        } else {

          console.log("CNPJ 2° digito verificador invalido!");

        }

      } else {

        return ZDK.Client.showAlert("CNPJ Inválido" + cnpjFormatado);

      }

    } else {

     return ZDK.Client.showAlert("CNPJ Inválido" + cnpjFormatado + " digitos" +
tamanhoCnpjFormatado);

  }
```

6- Versão 2JavaScript.

```
function verificaCnpj() {
  const campoCnpj =
document.getElementById("numeroCnpj").value;
  const cnpjFormatado =
campoCnpj.replaceAll(/[a-zA-z!@#$%^&*.-/////-]/g, "");

  const tamanhoCnpjFormatado =
cnpjFormatado.toString().length;

  if (cnpjFormatado == "null" || cnpjFormatado == "") {
    alert("CNPj inválido" + cnpjFormatado + " digitos"
+ tamanhoCnpjFormatado);
```

```
}

if (
  cnpjFormatado == "00000000000000" ||
  cnpjFormatado == "11111111111111" ||
  cnpjFormatado == "22222222222222" ||
  cnpjFormatado == "33333333333333" ||
  cnpjFormatado == "44444444444444" ||
  cnpjFormatado == "55555555555555" ||
  cnpjFormatado == "66666666666666" ||
  cnpjFormatado == "77777777777777" ||
  cnpjFormatado == "88888888888888" ||
  cnpjFormatado == "99999999999999"
) {
  alert(
    "Valor inválido de Cnpj!" +
      cnpjFormatado +
      " digitos" +
      tamanhoCnpjFormatado
  );
}

if (tamanhoCnpjFormatado == 14) {
  num1 = cnpjFormatado.substr(0, 1);
  num2 = cnpjFormatado.substr(1, 1);
  num3 = cnpjFormatado.substr(2, 1);
  num4 = cnpjFormatado.substr(3, 1);
  num5 = cnpjFormatado.substr(4, 1);
  num6 = cnpjFormatado.substr(5, 1);
  num7 = cnpjFormatado.substr(6, 1);
  num8 = cnpjFormatado.substr(7, 1);
  num9 = cnpjFormatado.substr(8, 1);
  num10 = cnpjFormatado.substr(9, 1);
  num11 = cnpjFormatado.substr(10, 1);
```

```
    num12 = cnpjFormatado.substr(11, 1);
    num13 = cnpjFormatado.substr(12, 1);
    num14 = cnpjFormatado.substr(13, 1);

    //_____________________Primeiro Digito
verificador_____________________________________
______

    num1Int = parseInt(num1);
    num2Int = parseInt(num2);
    num3Int = parseInt(num3);
    num4Int = parseInt(num4);
    num5Int = parseInt(num5);
    num6Int = parseInt(num6);
    num7Int = parseInt(num7);
    num8Int = parseInt(num8);
    num9Int = parseInt(num9);
    num10Int = parseInt(num10);
    num11Int = parseInt(num11);
    num12Int = parseInt(num12);
    num13Int = parseInt(num13);
    num14Int = parseInt(num14);

    calc1 = num1Int * 5;
    calc2 = num2Int * 4;
    calc3 = num3Int * 3;
    calc4 = num4Int * 2;
    calc5 = num5Int * 9;
    calc6 = num6Int * 8;
    calc7 = num7Int * 7;
    calc8 = num8Int * 6;
    calc9 = num9Int * 5;
    calc10 = num10Int * 4;
    calc11 = num11Int * 3;
```

```
    calc12 = num12Int * 2;

const somandoDigitos01 =
      calc1 +
      calc2 +
      calc3 +
      calc4 +
      calc5 +
      calc6 +
      calc7 +
      calc8 +
      calc9 +
      calc10 +
      calc11 +
      calc12;

    console.log(somandoDigitos01);

    divisao01 = somandoDigitos01 / 11;
    divisaoResto = somandoDigitos01 % 11;

    console.log("Divisão: " + divisao01);
    console.log("Resto da divisão:" + divisaoResto);
    const primeiroDigitoVerificador = 11 -
divisaoResto;

    //_____________________________Segundo digito
Verificador_____________________________________

    calc1 = num1Int * 6;
    calc2 = num2Int * 5;
    calc3 = num3Int * 4;
    calc4 = num4Int * 3;
    calc5 = num5Int * 2;
```

```
    calc6 = num6Int * 9;
    calc7 = num7Int * 8;
    calc8 = num8Int * 7;
    calc9 = num9Int * 6;
    calc10 = num10Int * 5;
    calc11 = num11Int * 4;
    calc12 = num12Int * 3;
    calc13 = num13Int * 2;

    const somandoDigitos02 =
      calc1 +
      calc2 +
      calc3 +
      calc4 +
      calc5 +
      calc6 +
      calc7 +
      calc8 +
      calc9 +
      calc10 +
      calc11 +
      calc12 +
      calc13;

    console.log(somandoDigitos02);

    divisao02 = somandoDigitos02 / 11;
    divisaoResto02 = somandoDigitos02 % 11;

    console.log("Divisão2: " + divisao02);
    console.log("Resto da divisão2:" + divisaoResto02);
    const segundoDigitoVerificador = 11 -
divisaoResto02;
```

```
    if (divisaoResto == 0 || divisaoResto == 1) {
      console.log("1° Digito verificador Válido =0");

      if (divisaoResto02 == 0 || divisaoResto02 == 1) {
        console.log("2° Digito verificador Válido =0");
        return alert("CNPJ Aceito");
      } else {
        console.log("CNPJ 2°Digito verificador
invalido!");
      }
    }

    if (divisaoResto <= 10 && divisaoResto >= 2 &&
primeiroDigitoVerificador==num13Int) {

      console.log(
        " O Primeiro digito verificador é: " +
primeiroDigitoVerificador
      );

      if (divisaoResto02 <= 10 && divisaoResto02 >= 2
&& segundoDigitoVerificador==num14Int) {

        console.log(
          " O segundo digito verificador é: " +
segundoDigitoVerificador
        );
        return alert("CNPJ Aceito");
      } else {
        console.log("CNPJ 2° digito verificador
invalido!");
      }
    } else {
      console.log("CNPJ Inválido" + cnpjFormatado);
```

```
    }
  } else {
    alert("CNPJ Inválido" + cnpjFormatado + " digitos"
+ tamanhoCnpjFormatado);
  }
}
```