

Itaú

- [Credenciais para API](#)
- [Dicas preenchimento convênio - Itaú](#)
- [Gerar certificado dinâmico Itaú](#)
- [Renovar certificado dinâmico Itaú](#)

Credenciais para API

1 - Para o Cliente

1.1 - Solicitar a criação de credenciais de produção

Entre em contato com o gerente de sua conta e/ou com os canais de suporte do Itaú para a criação do `client_id` e `client_secret`.

1.2 - Adquirir certificado dinâmico

Diferente dos demais bancos, o Itaú solicita um certificado especial, denominado "**Certificado dinâmico**", alguns gerentes de conta podem conhecer o procedimento e através do **certificado padrão** já gerarem o **certificado dinâmico** para você. Porém, caso seu gerente de conta não conheça o processo, [clique aqui para acessar a documentação de como gera-lo](#), ou então, [clique aqui para acessar a documentação de como renova-lo](#).

2 - Para a Microsys

No MsysGestor, preencher da seguinte forma:

Certificado Digital: Realizar o upload do **certificado dinâmico** para o sistema Microsys.

API ID:

Informar com o campo **Client_id**.

API SECRET: informar com o campo **client_secret**.

Dicas preenchimento convênio - Itaú

As informações abaixo são dicas para auxiliar o cadastro dos dados bancários, é de suma importância que todo e qualquer dado seja confirmado com o banco.

Cadastro de conta corrente

Campo	Dica
Tipo	Informar o tipo da conta. Pode ser "Conta corrente" ou "Conta poupança".
Banco	Informar o código do banco de acordo com a FEBRABAN. Para Itaú deverá ser o código 341.
Número da conta	Informar o número da conta . campo composto por 5 dígitos + DV . Exemplo: 12345-6
Agência	Informar a agência mantenedora da conta. Este campo é composto por 4 dígitos sem DV . Exemplo: 9999
Código beneficiário	Informar o número da conta sem o DV . Exemplo: 12345

Cadastro de convênio

Campo	Dica
CNAB	Layout adotado pelo banco. Pode ser "240" ou "400"
Número	Neste campo pode ser informado a agência mais o número da conta + DV. Exemplo: 9999123456
Descrição	Campo de texto para controle interno e identificação do convênio.
Número carteira	Informar o Número da carteira de acordo com o repassado pelo banco. Pode ser: 109 - Cobrança com registro 104 - Cobrança com registro (Escritural Eletrônica) 175 - Cobrança sem registro
Número da remessa	Informado o número da remessa atual.

Referência

<https://atendimento.tecnospeed.com.br/hc/pt-br/articles/360015115794-Banco-Ita%C3%BA>

Gerar certificado dinâmico Itaú

Requisitos

Para a geração do certificado dinâmico, é necessário ser instalado:

1. OpenSSL - Precisamos dele para rodar alguns comandos.
2. GitBash - Terminal utilizado para escrever os comandos.
3. Java - Necessário para rodar alguns scripts de código.
4. Postman - Software para acesso de endpoints.

1- Obter par de chaves

Para gerar um par de chaves RSA, a qual é composta por uma chave privada (para decifrar) e uma chave pública (para criptografar), execute o seguinte comando no **GitBash** :

```
openssl genpkey -out private.pem -algorithm RSA -pkeyopt rsa_keygen_bits:2048  
openssl rsa -in private.pem -pubout -out public.pem
```

Após executar os comandos, o par de chaves será gerado através de um arquivo denominado **public.pem** e outro **private.pem**.

Exemplo do conteúdo do arquivo **public.pem**:

```
-----BEGIN PUBLIC KEY-----MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwTcm21MHeGMHN8e4uAZiU/pKoIrtLoy0uoLmpogmWdiV20Jz30jysQf7JLE1o7xmkwHI8JGg41wuUv+CF6Ax0kk3EwV6AVkLnI2l0AY2LbrgwDtsGQh6gslIUJhiadHqwwnezBanTIgV+Z4sa020SWZGbWufXJ9PVP91LXy/d4oyHGbkMVUQlyK350+iL8Pw0reTy6MiKKPrdSaKkj87PhrE/Fd3vIE1ANqhJjIghNtB4iqo4VF8bBZKohkShu8IA5Y3uV20kMz3lD/jNtJn8J7D7uKYCB8Xy9Sy/80+rwvly7i7nWLG1zpVz7KD7+3KWrnmuf2o1gefCin1FLvpwIDAQAB-----END PUBLIC KEY-----
```

2 - Recebimento do Token temporário

Nesta etapa, deve-se enviar o arquivo public.pem para o time de atendimento do Itaú.

Com esse arquivo o time do Itaú vai criptografar as informações de "**Token temporário**" e "**client id**" e envia-lo a seu e-mail cadastrado junto a eles. Esse processo pode demorar em média 1 dia útil.

Importante ressaltar que o Token temporário tem validade de **7 dias**.

Exemplo de e-mail retornado pela equipe Itaú:

E-mail de credenciais

Olá parceiro, seja bem vindo ao Itaú_

Você já pode gerar seu certificado dinâmico com as credenciais abaixo:

ClientId: KWMMqDhJ+p/2fejfx2uktyDnAjT8B3u2FZCyPlyA5U5t3IU4VVcS6wY5KG+SZ0+a

Token Temporário:

+B8tD52L3qrexAjr3vinhj1FGgrfr/iGzx/lKhE2mHJPcj0CnLbCcqp52w3YDXNL+Zi2RKmEp0ZwRFzZYJzWNBmiQh65CKTIRMPBA/2

Chave Sessão:

KT9d4C4mthX9aFWHGwUlVBS3quLagK2NrDDAXxQWiBCcpu7RDdQE+/C0dATzAjMUUiqsuKpFSnakhI7b0uNAF3CzqNgTnGc6xtu3i0

3 - Decifrar credenciais

Após receber o e-mail com as informações, crie um arquivo chamado **Criptografia.java** com o seguinte código dentro dele:

```
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.InputStream;
import java.io.InputStreamReader;import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.KeyFactory;
import java.security.NoSuchAlgorithmException;import java.security.interfaces.RSAPrivateKey;
import java.security.spec.PKCS8EncodedKeySpec;
import java.util.Base64;
import java.util.Scanner;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

public class Criptografia {
    private static String extrairChaveRsaPem(String tipoChave, String arquivoChavesRsa)
    {
        try {
            InputStream is = new FileInputStream(arquivoChavesRsa);
            @SuppressWarnings("resource")    BufferedReader br = new BufferedReader(new
            InputStreamReader(is));
            StringBuilder sb = new StringBuilder();                boolean inKey =
            false;                for (String line = br.readLine(); line != null; line = br.readLine())
            {
                if (!inKey) {                    if (line.startsWith("-----BEGIN ") &&
                line.endsWith(" " + tipoChave + " KEY-----")) {                        inKey =
                true;
                }
            }
        }
    }
}
```



```

        } else {
            if (line.startsWith("-----END ") &&
line.endsWith(" " + tipoChave + " KEY-----")) {
                inKey =
false;

                break;
            }
            sb.append(line);
        }
    }
    return sb.toString();
}
catch(Exception e) {
    e.printStackTrace();
}
return null;
}

public static String decriptografiaAes(SecretKey key, String cipherText) {
    try {
        Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5Padding");
        IvParameterSpec iv = new
IvParameterSpec(new byte[16]);
        cipher.init(Cipher.DECRYPT_MODE, key,
iv);
        byte[] plainText =
cipher.doFinal(Base64.getDecoder().decode(cipherText));
        return new String(plainText);
    }
    catch(NoSuchAlgorithmException e) {
        e.printStackTrace();
    }
    catch(BadPaddingException e) {
        e.printStackTrace();
    }
    catch(IllegalBlockSizeException e) {
        e.printStackTrace();
    }
    catch(InvalidKeyException e) {
        e.printStackTrace();
    }
    catch(InvalidAlgorithmParameterException e) {

```

```

        e.printStackTrace();
    }
    catch(NoSuchPaddingException e) {
        e.printStackTrace();
    }
    return null;
}

private static byte[] decriptografiaRsa(String caminhoChavePrivada, String
dadosCifrados) {
    try {
        String chavePrivada = extrairChaveRsaPem("PRIVATE",
caminhoChavePrivada);

        PKCS8EncodedKeySpec spec = new
PKCS8EncodedKeySpec(Base64.getDecoder().decode(chavePrivada.toString()));
        KeyFactory kf = KeyFactory.getInstance("RSA");
        RSAPrivateKey privateKey =
(RSAPrivateKey) kf.generatePrivate(spec);

        Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
        cipher.init(Cipher.DECRYPT_MODE, privateKey);

        return cipher.doFinal(Base64.getDecoder().decode(dadosCifrados));
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}

public static void main(String[] args)
{
    SecretKey chaveSessao = null;
    Scanner scanner = new
Scanner(System.in);

    try {
        System.out.println("\n=====");
        System.out.println("Informe as informações recebidas no e-mail");
        System.out.println("=====");

        System.out.println("\nClientId: ");
        System.out.flush();
        String clientIdCifrado =
scanner.nextLine().trim();

        System.out.println("\nToken Temporário: ");
        System.out.flush();
    }
}

```

```

        String tokenCifrado = scanner.nextLine().trim();
        System.out.println("\nChave Sessão: ");
        System.out.flush();
        String chaveSessaoCifrada = scanner.nextLine().trim();
        System.out.println("\nCaminho chave privada: ");
        System.out.flush();
        String caminhoChavePrivada = scanner.nextLine().trim();
        System.out.flush();
        scanner.close();
        System.out.println("\n=====");
        System.out.println("    Processo de Decriptografia    ");
        System.out.println("=====");
        // Decifra a chave de sessao AES com a chave RSA privada
        byte[] chaveSessaoDecifrada = decriptografiaRsa(caminhoChavePrivada,
        chaveSessaoCifrada);
        chaveSessao = new SecretKeySpec(chaveSessaoDecifrada, 0,
        chaveSessaoDecifrada.length, "AES");
        // Decriptografa a credencial atraves da
        chave de sessao AES
        String clientIdDecifrada = decriptografiaAes(chaveSessao,
        clientIdCifrado);
        System.out.println("\nClient id decifrado com a chave de sessao
        AES:\n[ " + new String(clientIdDecifrada) + " ]");
        String tokenDecifrado = decriptografiaAes(chaveSessao, tokenCifrado);
        System.out.println("\nToken
        decifrado com a chave de sessao AES:\n[ " + new String(tokenDecifrado) + " ]");
    }
    catch(Exception e) {
        e.printStackTrace();
    }
}
}

```

Após isso, execute os seguintes comandos no terminal **GitBash** (O terminal deve ser aberto na mesma pasta que o arquivo **Criptografia.java** foi salvo):

```
javac Criptografia.java
java -cp . Criptografia
```

Ao executar esses comandos, preencha os dados solicitados no terminal **GitBash** de acordo com o que foi recebido no e-mail:

ClientId: ClientId presente no e-mail

Token Temporário: Token Temporário presente no e-mail

Chave sessão: Chave sessão presente no e-mail

Caminho da chave privada: escreva o caminho onde está o arquivo private.pem gerado anteriormente.

Após informar esses dados, o terminal deve exibir a resposta, semelhante as informações abaixo:

```
Credenciais decifradas
Processo de Decriptografia
Client id decifrada com a chave de sessao AES:
e26f2f89-0ead-4ca6-8bc3-dd44b4ab3cc7
Token decifrado com a chave de sessao AES:
eyJraWQiOiJhZTYxYWl0yNTRhLTQ5ZWQtd0MzNC05ZDJlN2E0MzZiNGQuaG9tLmdlbi4xNTk3NjAwMzM2OTkyLmp3dCIsImFsZy...
```

4 - Geração do arquivo .csr e .key

Para gerar os arquivos **.csr** e **.key**, deve criar uma pasta para salvar os arquivos, e em seguida abrir o terminal **GitBash** nesta pasta.

Em sequência rodar o seguinte comando:

Para Windows:

```
openssl req -new -subj "//CN={{CLIENT_ID}}\OU=SITE OU APP DO PARCEIRO\L=CIDADE\ST=ESTADO\C=BR" -out ARQ
```

Para Linux:

```
openssl req -new -subj "/CN={{CLIENT_ID}}/OU=SITE OU APP DO PARCEIRO/L=CIDADE/ST=ESTADO/C=BR" -out ARQ
```

Lembre-se de alterar algumas das informações do script acima seguindo os dados repassados pelo banco:

CN= ClientID disponibilizado no e-mail

OU= Deixe preenchido com o nome do app que vai utilizar o certificado (exemplo: **MSYSGESTOR**)

L= Cidade onde se localiza a Agência do Cliente (exemplo: **SAO MIGUEL DO OESTE**)

ESTADO= Estado onde se localiza a Agência do Cliente (exemplo: **SANTA CATARINA**)



C= País onde se localiza a Agência do Cliente (exemplo: **BR**)

ARQUIVO_REQUEST_CERTIFICADO.csr= Nome do certificado, deve manter o ".csr" no final

ARQUIVO_CHAVE_PRIVADA.key=Nome da chave vinculada ao certificado, deve manter o ".key" no final

OBS: as informações **OU**, **L**, **ESTADO** e **C** devem ser todas escritas em caixa alta e sem nenhuma acentuação ou caractere especial, segundo os exemplos acima.

Ao executar o comando, os seguintes arquivos serão gerados na pasta:

Nome	Data de modificação	Tipo	Tamanho
 certificado.csr	18/02/2022 15:13	Arquivo CSR	2 KB
 certificadoKey.key	18/02/2022 15:13	Arquivo KEY	2 KB

5 - Autenticar arquivo .csr e .key em um arquivo .crt

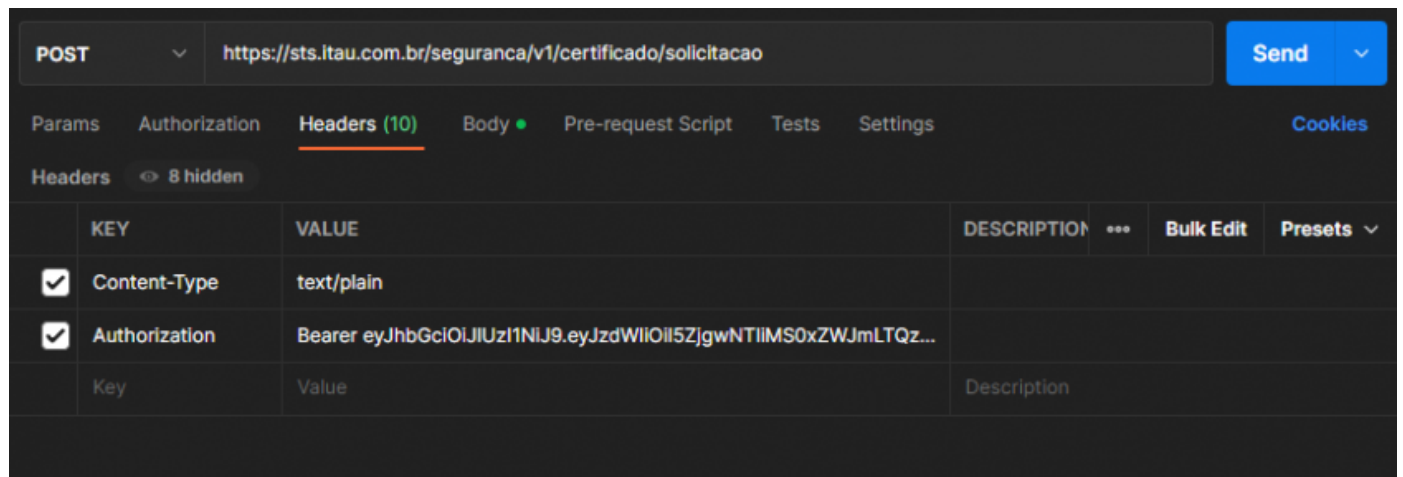
O envio do arquivo **.csr** para o banco deve ser feito via API, através de um *endpoint POST*. Aconselhamos o uso do software **Postman** para isso.

Na **URL** da requisição:

- inserir **https://sts.itaubr.com.br/seguranca/v1/certificado/solicitacao**

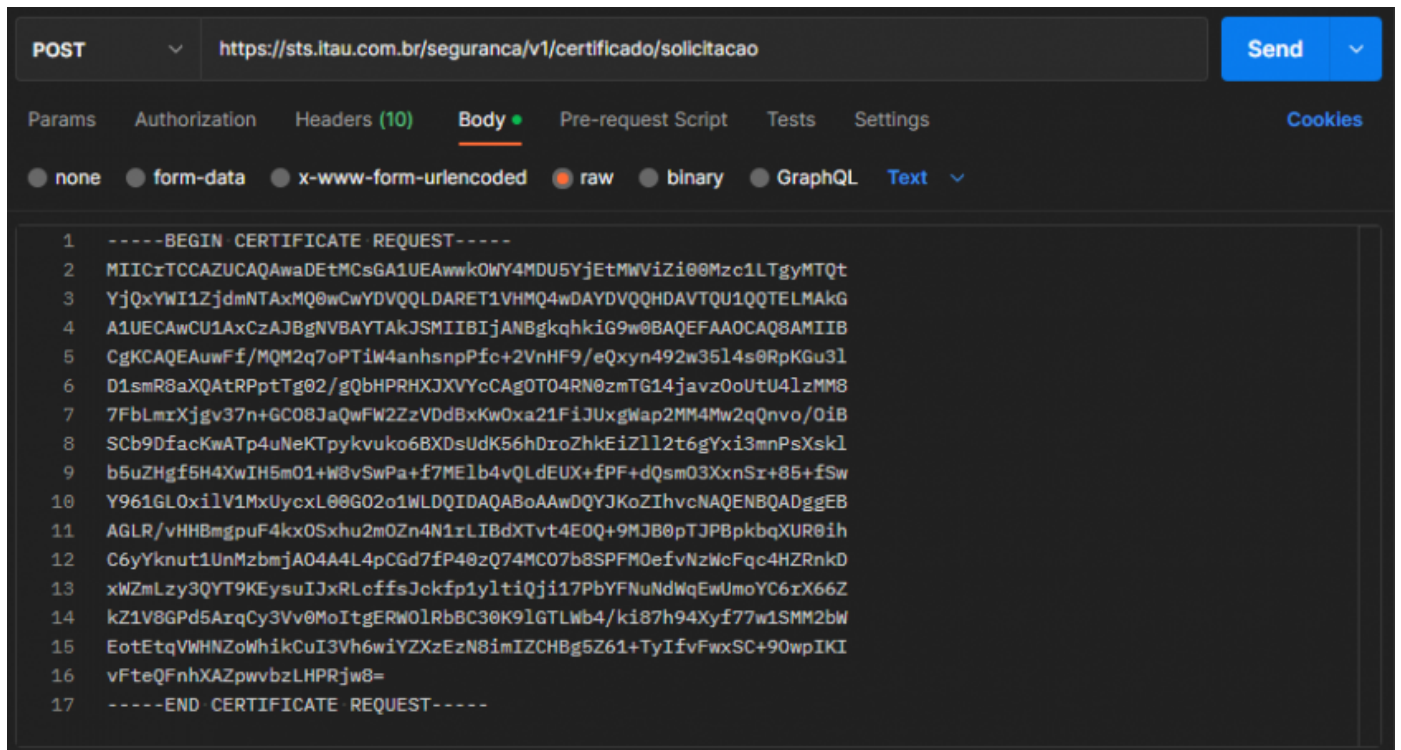
No **header** da requisição:

- Insira um Content-Type= text/plain
- Insira um Authorization: bearer {token temporário}



No **body** da requisição:

- insira o **certificado.csr** gerado anteriormente em texto. Para adquirir o texto desse arquivo, é só abri-lo com um editor de texto.



Ao enviar essa requisição POST para o Itaú, vai ser retornada duas informações, o Secret e a nova informação do certificado.

Copie o Secret e salve em um arquivo com extensão **.key** e o conteúdo do certificado em um arquivo com extensão **.crt**, preferencialmente salve ambos em uma nova pasta.

6 - Gerar arquivo .pfx

Acesse a pasta criada com os arquivos **.crt** e **.key** gerados no passo anterior, e após isso abra o **GitBash** nessa pasta.

Insira a seguinte informação no terminal:

```
openssl pkcs12 -export -out certificado_dinamico_itau.pfx -inkey certificadoKey.key -in certificadoGerado.crt
```

Lembre-se de alterar algumas das informações do script acima seguindo os dados repassados pelo banco:

certificadoKey.key

= Insira o nome do arquivo gerado no passo anterior, mantenha a extensão **.key** no final.

certificadoGerado.crt= Insira o nome do arquivo gerado no passo anterior, mantenha a extensão **.crt** no final.

Em sequência vai ser solicitado uma senha para o certificado e a confirmação da senha. Preencha a senha desejada as duas vezes.

Ao final vai ser gerado o seu **certificado dinâmico** para realizar o upload dele no sistema **MsysGestor**.

Referências

<https://atendimento.tecnospeed.com.br/hc/pt-br/articles/7003805388567-Utilizando-o-registro-via-Web-Service-Banc%C3%A1rio-com-o-Ita%C3%BA-V2->

<https://forum.casadodesenvolvedor.com.br/topic/43985-gerando-certificado-crt-para-o-banco-ita%C3%BA-pix-e-boleto-webservice/>

<https://forum.casadodesenvolvedor.com.br/topic/43793-como-converter-um-arquivo-de-certificado-crt-para-o-formato-pfx/>

<https://devportal.itau.com.br/certificado-dinamico>

Renovar certificado dinâmico Itaú

O certificado dinâmico tem um prazo para expirar de 12 meses a partir do momento de sua criação. Caso expire, deve ser feito todos os passos descritos no artigo "[Gerar certificado dinâmico Itaú](#)" novamente.

O Itaú permite a renovação do certificado dinâmico a partir dos 60 dias que antecedem o término da validade do certificado antigo. Veremos os passos para esse processo a seguir:

Requisitos

1. OpenSSL - Precisamos dele para rodar alguns comandos.
2. GitBash - Terminal utilizado para escrever os comandos.
3. Postman - Software para acesso de endpoints.

1 - Geração do arquivo .csr e .key

Para gerar os arquivos **.csr** e **.key**, deve criar uma pasta para salvar os arquivos, e em seguida abrir o terminal **GitBash** nesta pasta.

Em sequência rodar o seguinte comando:

Para Windows:

```
openssl req -new -subj "//CN={{CLIENT_ID}}\OU=SITE OU APP DO PARCEIRO\L=CIDADE\ST=ESTADO\C=BR" -out ARQ
```

Para Linux:

```
openssl req -new -subj "/CN={{CLIENT_ID}}/OU=SITE OU APP DO PARCEIRO/L=CIDADE/ST=ESTADO/C=BR" -out ARQ
```

Lembre-se de alterar algumas das informações do script acima seguindo os dados repassados pelo banco:

CN= ClientID disponibilizado no e-mail

OU= Deixe preenchido com o nome do app que vai utilizar o certificado (exemplo: **MSYSGESTOR**)

L= Cidade onde se localiza a Agência do Cliente (exemplo: **SAO MIGUEL DO OESTE**)

ESTADO= Estado onde se localiza a Agência do Cliente (exemplo: **SANTA CATARINA**)



C= País onde se localiza a Agência do Cliente (exemplo: **BR**)

ARQUIVO_REQUEST_CERTIFICADO.csr= Nome do certificado, deve manter o ".csr" no final

ARQUIVO_CHAVE_PRIVADA.key=Nome da chave vinculada ao certificado, deve manter o ".key" no final

OBS: as informações **OU**, **L**, **ESTADO** e **C** devem ser todas escritas em caixa alta e sem nenhuma acentuação ou caractere especial, segundo os exemplos acima.

Ao executar o comando, os seguintes arquivos serão gerados na pasta:

Nome	Data de modificação	Tipo	Tamanho
 certificado.csr	18/02/2022 15:13	Arquivo CSR	2 KB
 certificadoKey.key	18/02/2022 15:13	Arquivo KEY	2 KB

2 - Configurando a coleção Postman Itaú

Para os próximos passos, torna-se necessário uma comunicação com a API do Itaú, a forma mais simples de realizar essa comunicação é através do software **Posman**. O Itaú disponibiliza uma coleção para ser utilizada no software, essa coleção serve para facilitar o processo, pois traz grande parte dos dados necessários já preenchidos e/ou configurados.

2.1 Baixar a coleção

acesse o site <https://devportal.itau.com.br/certificado-dinamico#criacao-ativacao> e baixe a coleção.

Criação e ativação do certificado dinâmico via Postman

Disponibilizamos uma collection para você realizar a criação e ativação do certificado dinâmico.



Faça o download da nossa [Collection](#) do Postman e comece agora mesmo.

Como enviar o certificado dinâmico via Postman

Siga as instruções abaixo para importar o **arquivo_certificado.crt** e **arquivo_chave_privada.key**, que foram gerados no processo de ativação do certificado dinâmico. No Postman basta acessar as configurações e, na guia de certificados, realizar o procedimento como exemplificado na imagem abaixo.



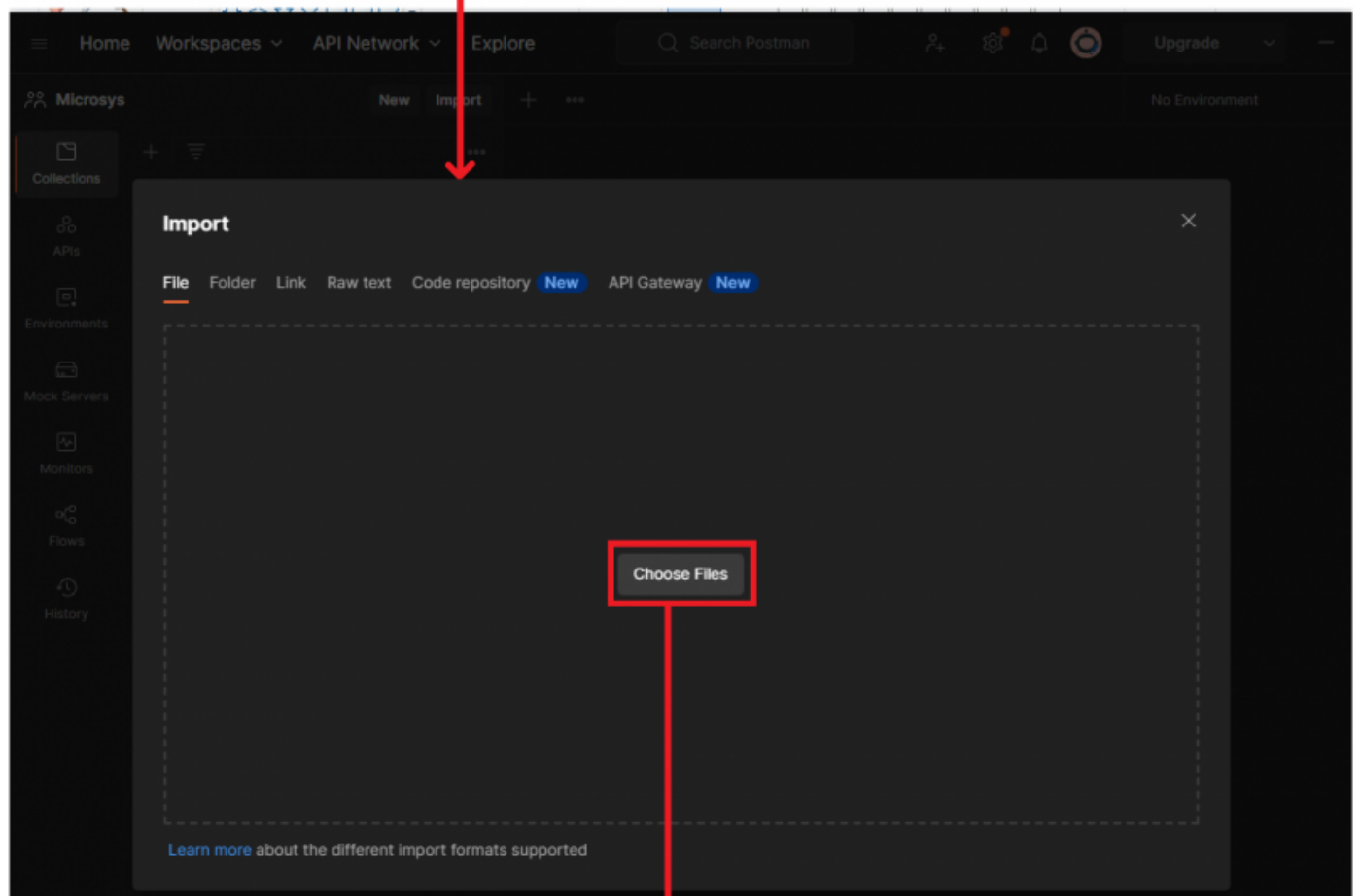
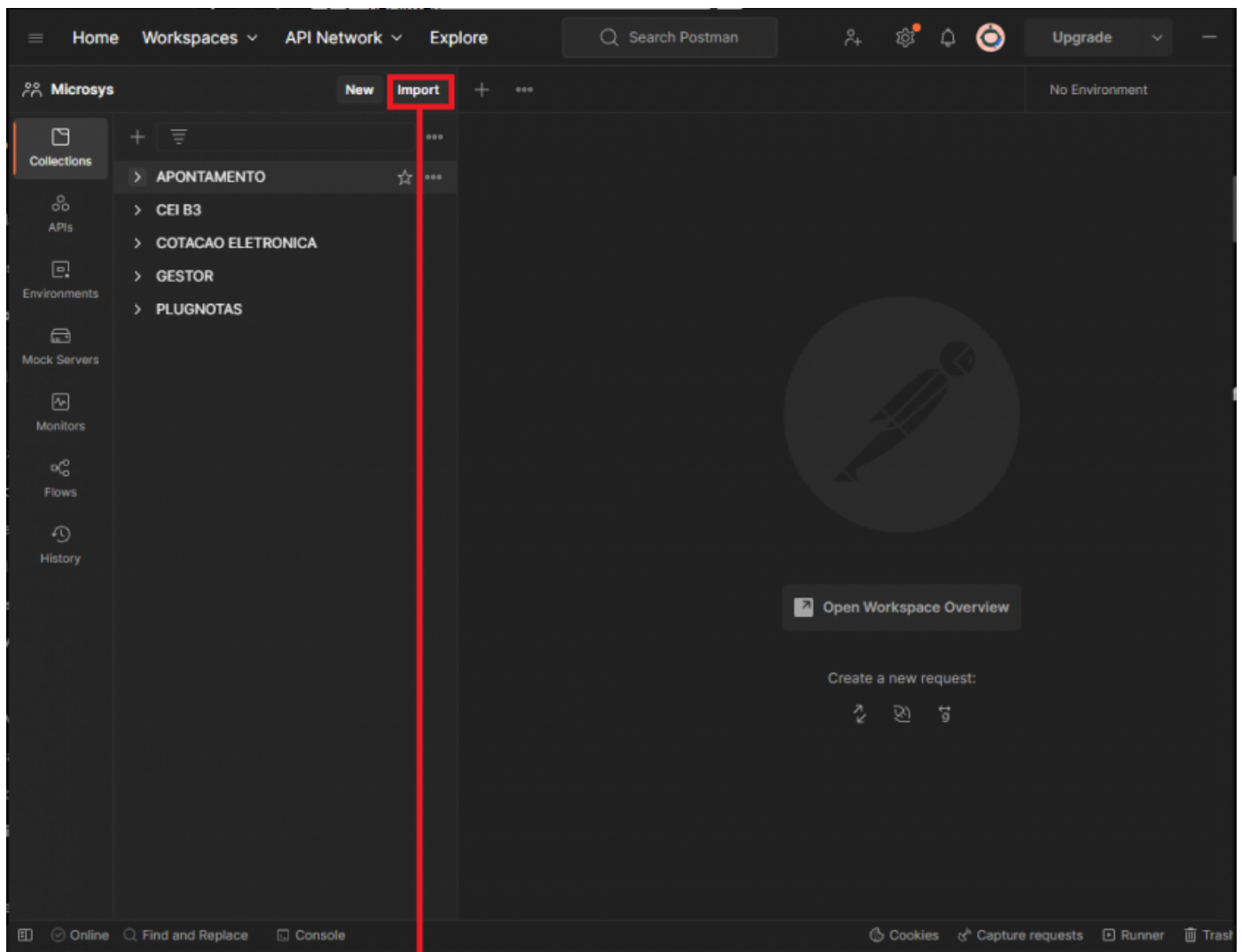
Lembre-se que em alguns processos relacionados ao certificado, você utilizará o host **api.itau.com.br** e em outros, **sts.itau.com.br**. Se estiver em dúvida de qual utilizar, entre em contato com o seu ponto focal no Itaú.

Para configurar o certificado no Postman, execute as etapas a seguir:

certificado_dinamic....zip ^

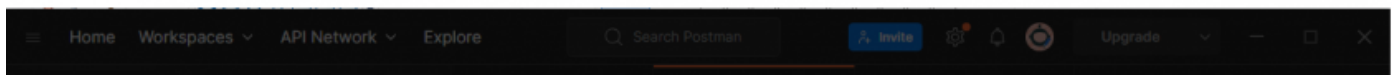
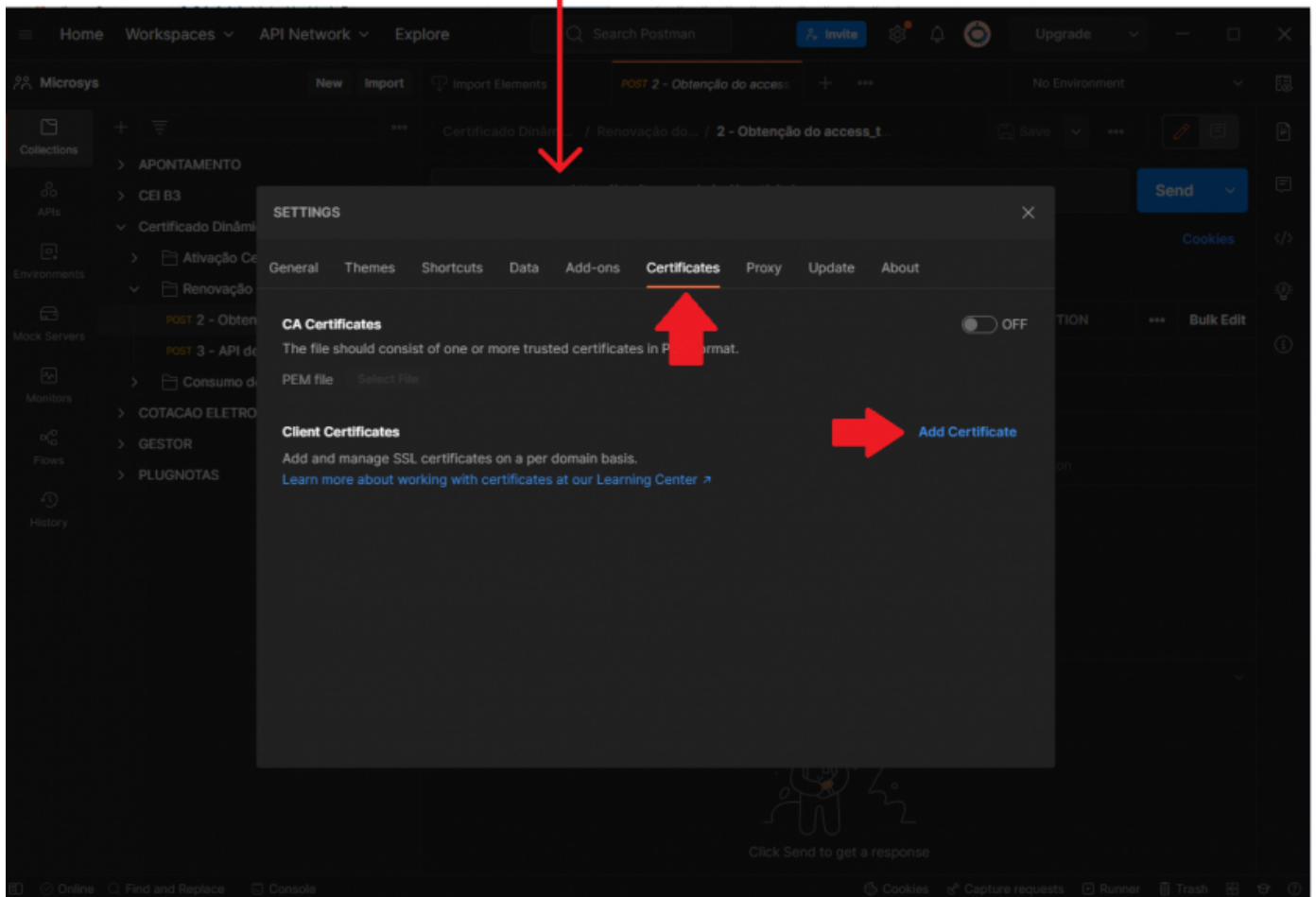
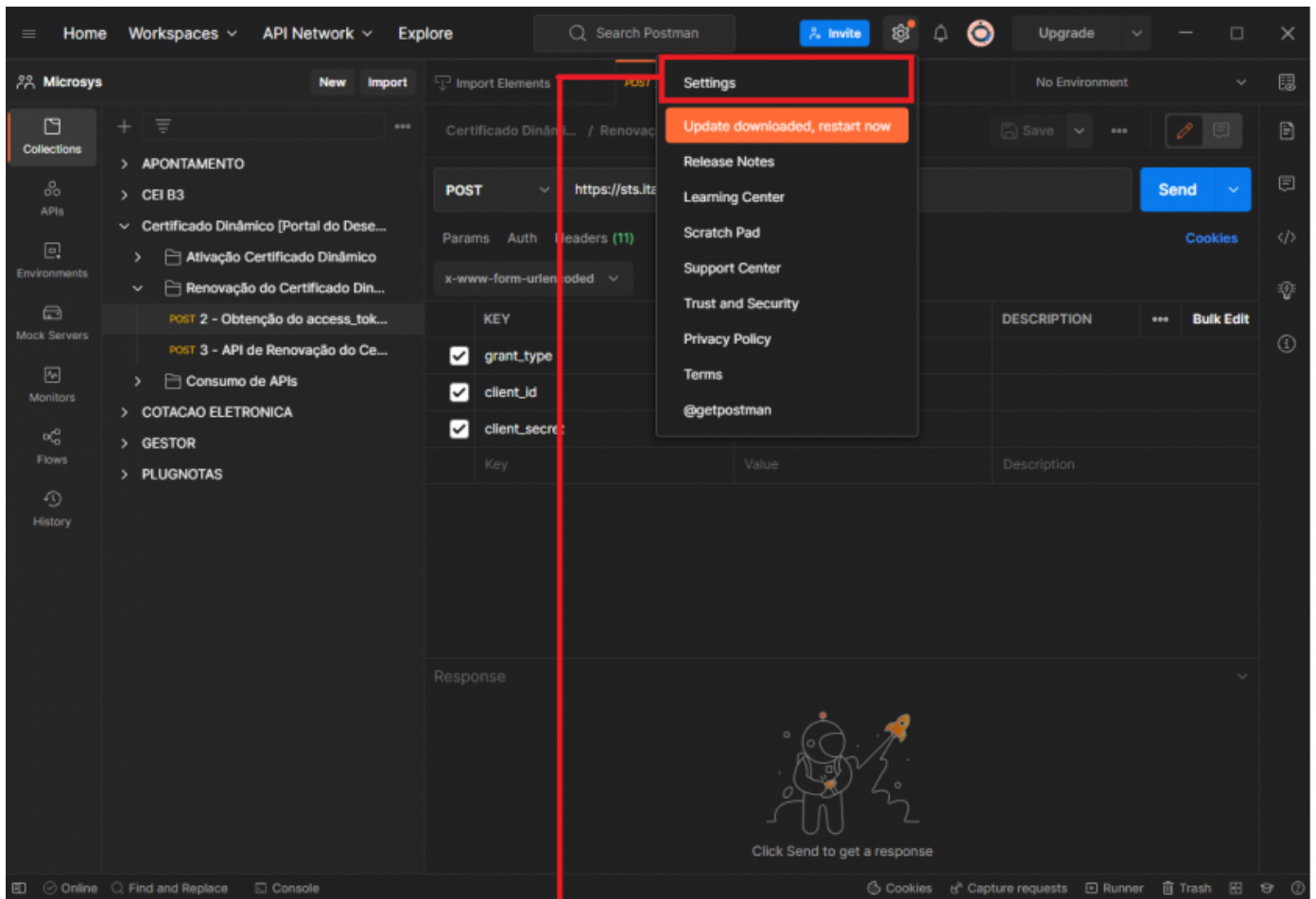


2.2 - Importe a coleção para o Postman



3 - Gerando access token para a renovação do certificado

3.1 - Informar o .crt a vencer para autenticação



Host: Preencher com sts.itau.com.br

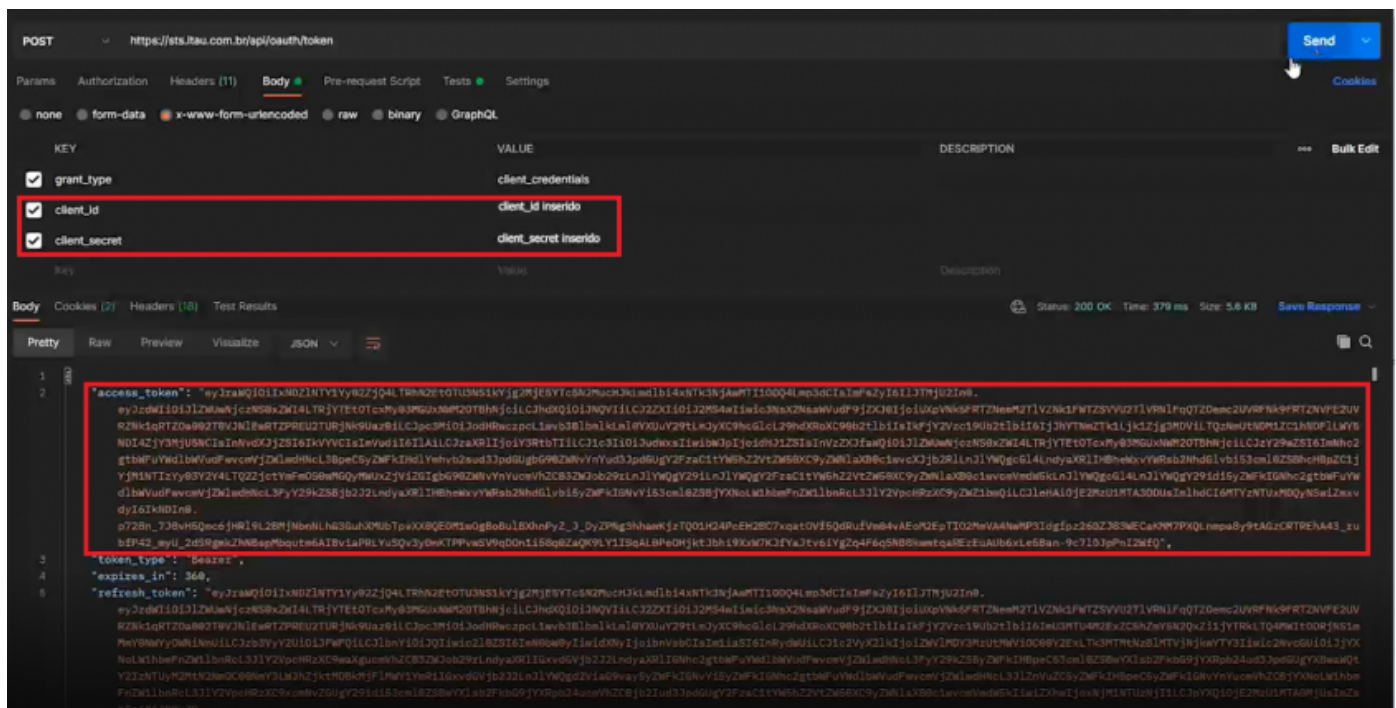
CRT file: Inserir o certificado a vencer no formato **.crt**

KEY file: Inserir o arquivo **.key** do certificado a vencer

3.2 - enviando requisição POST para gerar o Access token

Após vincular o certificado ao Postman, acessar o endpoint importado da coleção referente a "**Obtenção do access_token**".

Neste endpoint, preencher o body, com o **client_id** e **client_secret** repassado pelo banco quando foi criado o primeiro certificado dinâmico. e clicar em "Send". Após isso será retornado o access_token:



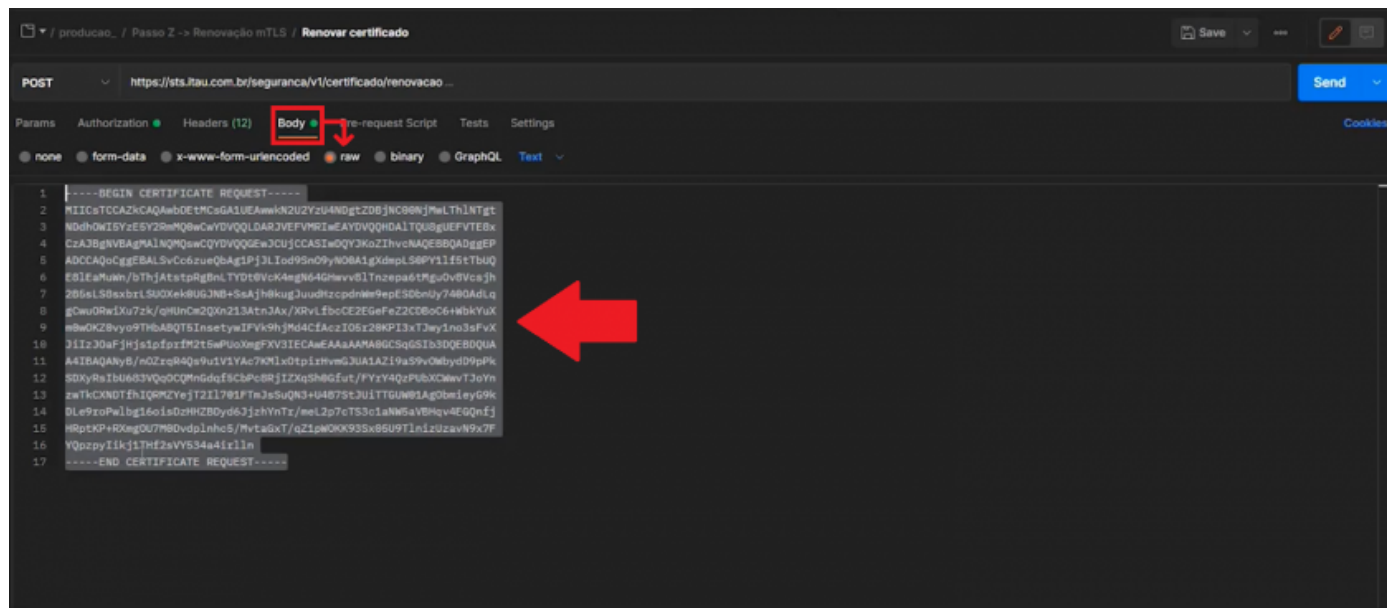
4 - Solicitar renovação de certificado

Nesta etapa, deve-se acessar o endpoint importado da coleção referente a "**Renovação do certificado dinâmico**", e incluir na aba **Authorization**:

- **Type:** Bearer Token

- **Token:** Informar o `access_token` copiado na etapa anterior

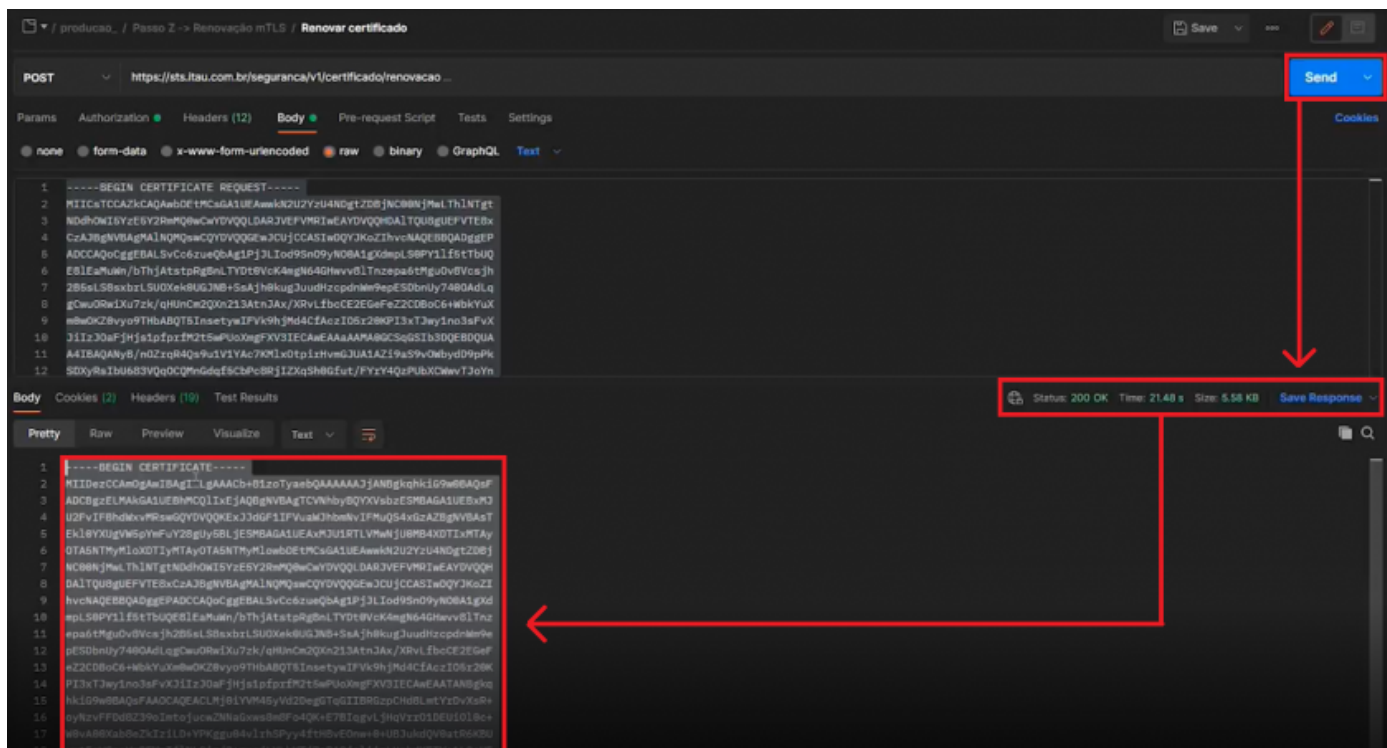
Já na aba **Body**, copiar o conteúdo do **.csr** gerado na **etapa 1** e colar na opção "**raw**":



Após isso, clique em "**Send**" para enviar a requisição.

5- Gerando certificado no formato **.crt**

Em caso de sucesso da etapa anterior, o Itaú vai retornar um novo conteúdo do certificado autorizado pelo banco.



Copie o conteúdo da resposta do Itaú, cole em um editor de texto no qual você esteja familiarizado e salve o arquivo no formato **.crt**, preferencialmente salve o novo certificado junto aos arquivos **.csr** e **.key**.

6 - Gerar arquivo .pfx

Acesse a pasta criada com os arquivos gerados no passo anterior, e após isso abra o **GitBash** nessa pasta.

Insira a seguinte informação no terminal:

```
openssl pkcs12 -export -out certificado_dinamico_itau.pfx -inkey certificadoKey.key -in certificadoGerado.crt
```

Lembre-se de alterar algumas das informações do script acima seguindo os dados repassados pelo banco:

certificadoKey.key= Insira o nome do arquivo gerado no passo anterior, mantenha a extensão **.key** no final.

certificadoGerado.crt

= Insira o nome do arquivo gerado no passo anterior, mantenha a extensão **.crt** no final.

Em sequência vai ser solicitado uma senha para o certificado e a confirmação da senha. Preencha a senha desejada as duas vezes.

Ao final vai ser gerado o seu **certificado dinâmico** para realizar o upload dele no sistema **MsysGestor**.

Referências

<https://devportal.itau.com.br/certificado-dinamico#renovacaoCertificado>

<https://forum.casadodesenvolvedor.com.br/topic/43793-como-converter-um-arquivo-de-certificado-crt-para-o-formato-pfx/>