

OUTUBRO DE 2024



# MANUAL DO DESENVOLVEDOR, v6.0

COMO UTILIZAR AS OPEN APIS DO BRADESCO

API STUDIO  
BANCO BRADESCO  
<https://banco.bradesco>

# ÍNDICE

<b>OBJETIVO .....</b>	<b>4</b>
<b>AUTENTICAÇÃO .....</b>	<b>5</b>
<b>1. Criação de um Client ID .....</b>	<b>5</b>
<b>AUTORIZAÇÃO .....</b>	<b>7</b>
<b>1. Gerando token de acesso em homologação .....</b>	<b>7</b>
<b>1.1 Criando um JWT assinado .....</b>	<b>7</b>
a) <b>Header.....</b>	<b>8</b>
b) <b>Payload.....</b>	<b>9</b>
c) <b>Assinatura .....</b>	<b>10</b>
<b>1.2 JWT convertido em base64.....</b>	<b>11</b>
<b>1.3 A Chamada do token.....</b>	<b>11</b>
<b>CONSUMO DE UMA API APÓS GERAÇÃO DO BEARER TOKEN .....</b>	<b>14</b>
<b>1. Consumo em homologação.....</b>	<b>14</b>
<b>2. Consumo em produção.....</b>	<b>19</b>
<b>TEMPLATES PARA CONTATO COM O SUPORTEAPI.....</b>	<b>20</b>
<b>1. Cadastro de um client ID.....</b>	<b>20</b>
<b>2. Renovações.....</b>	<b>20</b>
<b>3. Solicitação de Suporte Técnico e dúvidas .....</b>	<b>21</b>
<b>GUIA DE REFERÊNCIA .....</b>	<b>23</b>
<b>1. Criando um certificado auto assinado .....</b>	<b>23</b>
<b>GLOSSÁRIO .....</b>	<b>24</b>
<b>Open API .....</b>	<b>24</b>
<b>Protocolo mTLS.....</b>	<b>24</b>
<b>Certificado digital .....</b>	<b>24</b>
<b>OAuth 2.0 .....</b>	<b>24</b>
<b>Autenticação.....</b>	<b>24</b>
<b>Autorização .....</b>	<b>24</b>
<b>Client ID .....</b>	<b>24</b>
<b>Client Key .....</b>	<b>24</b>
<b>JWT .....</b>	<b>24</b>
<b>Base64 .....</b>	<b>24</b>
<b>Header.....</b>	<b>24</b>
<b>Requisição .....</b>	<b>25</b>

<b>Endpoint.....</b>	<b>25</b>
<b>Chave pública e privada.....</b>	<b>25</b>
<b>JWS .....</b>	<b>25</b>
<b>Access Token.....</b>	<b>25</b>
<b>Body .....</b>	<b>25</b>
<b>Header.....</b>	<b>25</b>
<b>URL.....</b>	<b>25</b>
<b>Payload.....</b>	<b>25</b>
<b>Query Parameters.....</b>	<b>25</b>



---

## OBJETIVO

O presente manual demonstra como as **Open APIs** do Banco Bradesco devem ser utilizadas. Nele você verá como o processo de **autenticação** e **autorização** deve ser realizado, quais os passos para criar cada um dos artefatos necessários para consumo das APIs, além de um glossário com os termos técnicos presentes no documento.



## AUTENTICAÇÃO

O acesso às **Open APIs** do Bradesco é feito através do uso alguns protocolos e padrões técnicos para garantir a segurança no tráfego das informações. Sempre que for necessário consumir alguma API do Bradesco, é necessário realizar o processo de **autorização**.

Nesse processo, o sistema que quer se conectar em nossas APIs precisa ser previamente conhecido pelo Bradesco. Para isso é necessário seguir os passos abaixo:

### 1. Criação de um Client ID

Para solicitar a criação de um **Client ID (equivalente a Client Key)**, é necessário enviar um e-mail para [suporte.api@bradesco.com.br](mailto:suporte.api@bradesco.com.br) seguindo o *template* "**Cadastro de Client ID**". Lembrando, **todos os e-mails enviados deverão possuir algum representante do banco em cópia**, esse representante pode ser o seu gerente de relacionamento ou implementador, por exemplo.

Esse template deve seguir o padrão abaixo:

Para **cadastro** de credenciais e certificado de novos parceiros em ambiente de **homologação**

- O assunto do e-mail deve seguir o padrão:
- CAD-HML | NOME DA API | RAZAO SOCIAL | CNPJ
- exemplo: CAD-HML | PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000-00
- Anexar o certificado (chave pública) no formato .cer, .crt ou .pem, **com validade mínima de 4 meses e máxima de 3 anos**. Lembrando que, **para o ambiente de homologação (teste), o certificado deverá ser um auto assinado**.
- Para ambiente de homologação serão aceitos **SOMENTE** certificados **auto assinados**.

**Em caso de dúvidas de como gerar um certificado auto assinado, consulte o nosso guia de referência "Criando um certificado auto assinado", presente no final deste manual.**

**O nome da API citado no exemplo anterior, deverá ser o nome do serviço a ser**

**consumido**, ou seja, caso o cliente deseje consumir um serviço de cobrança, por exemplo, o modelo deverá ser:

- CAD-HML | **COBRANÇA** | RAZÃO SOCIAL | CNPJ

Como toda comunicação de APIs server-to-server do Bradesco com parceiros é feita utilizando o **protocolo TLS**, no e-mail a ser enviado é necessário incluir um **certificado digital**.

Existem algumas regras específicas para os certificados digitais aceitos pelo Bradesco, sobretudo, no ambiente **produtivo**. Confira abaixo:

- Deve seguir o **padrão ICP-BRASIL** do tipo **A1**;
- **Tamanho mínimo** de 2048 bits;
- Utilizar algum **algoritmo** RSA como o RSASHA 256, 384 ou 512;
- **Data de validade/expiração** deve ser superior há 3 meses e no máximo 3 anos;
  - Data de validade a contar da **data da solicitação** de cadastramento ou renovação

Devido ao uso do protocolo TLS, é necessário que a comunicação seja feita usando um dos algoritmos/cifras abaixo, e com a versão 1.2 ou superior:

- TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256, e/ou
- TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384

Se sua solicitação estiver dentro dos requisitos definidos pelo banco, em até 3 dias úteis forneceremos um **Client ID / Client Key** de acesso.

Com o **Client ID / Client Key** será possível sua aplicação provar que possui autenticidade para usar nossas APIs. O próximo passo é garantir que existe autoridade para usar determinada API, esse passo será feito através da geração de um dado chamado **Access Token** ou **token JWT**.

## AUTORIZAÇÃO

Assim como especificado pelo padrão **OAuth 2.0**, sempre que você quer consumir uma API é necessário provar que sua aplicação tem autorização e autenticidade para usá-la.

Para provar que existe **autorização** para acessar uma API, é necessário obter um **token JWT** através do nosso serviço de autorização, nos próximos tópicos você aprenderá desde obter o **token JWT** a também consumir uma API de teste no nosso ambiente de Homologação.

Esse **token** traz diversas informações sobre sua aplicação, codificadas em **Base64** e serão usadas nas chamadas feitas para a API que você irá consumir.



## 1. Gerando token de acesso em homologação (*access token/token JWT*)

Para acessar uma API do Bradesco no ambiente de **homologação**, você deve gerar o **token** a partir da **URL** abaixo:

- <https://proxy.api.prebanco.com.br/auth/server/v1.1/token>

As requisições de geração de **token** deverão conter um **JWS** no **body**. Um JWS, de forma direta, é um JWT assinado com uma chave privada. No seu caso, a chave privada será pertencente a chave pública que você nos enviou no início do processo do cadastro, assim, ambas formarão um par de chaves que poderão validar a sua autoridade.

### 1.1 Criando um JWT assinado (JWS)

O JWT é uma estrutura JSON formada por três partes: um **header**, um **payload** e uma assinatura da concatenação do header+payload separados por ponto. Cada um dos três itens são uma estrutura JSON apartada. Existe um padrão para cada um desses campos e como usá-los corretamente, será descrito a seguir:

#### a. Header

Objeto JSON contendo os parâmetros que descrevem as operações criptográficas e os parâmetros empregados. O cabeçalho JOSE (JSON Object Signing and Encryption) é composto por um conjunto de parâmetros de cabeçalho que normalmente consistem em um par nome/valor: o algoritmo de hash sendo usado (por exemplo, HMAC SHA256 ou RSA) e o tipo do JWT. No nosso caso, o Header do nosso JWT seguirá o modelo abaixo:

```
{
  "alg": "RS256",
  "typ": "JWT"
}
```



## b. Payload

O **payload** contém campos que também são conhecidos como *claims*. Esses campos incluem declarações de segurança verificáveis, como a identidade do usuário, tempo e as permissões.

```
{
  "aud": "https://proxy.api.prebanco.com.br/auth/server/v1.1/token",
  "sub": "suaclientkey",
  "iat": 1702580245,
  "exp": 1702583845,
  "jti": 1702580245000,
  "ver": "1.1"
}
```

- O campo **aud** sempre possuirá a **URL** de **token** completa que está sendo utilizada para o ambiente que você está realizando a sua requisição.
- O campo **sub** será a sua clientkey (client Id), a mesma que nós do suporteAPI enviamos por e-mail após o cadastramento no nosso ambiente.
- O campo **iat** será o horário atual em **segundos**. Sempre que for gerar um **novo token**, é necessário atualizá-lo.
- O campo **exp** é o mesmo **"iat"** porém, 1hr a frente, ou 3600 somado ao número do **"iat"**. Sempre que for gerar um **token**, é necessário atualizá-lo também.
- O campo **jti** nada mais é do que o **"iat"** em **milissegundos** e **não** segundos. Para simplificar, você pode reaproveitar o **"iat"** e concatenar três zeros no final do valor, assim ele se tornará um **"jti"** válido.

### c. Assinatura

Para assinar um JWT e transformá-lo em um JWS, primeiro precisamos transformar o **header** e o **payload** vistos anteriormente em Base64 e concatená-los com um ponto (.)

Um exemplo:

O **header** e **payload** abaixo quando convertidos em base 64 e separados por um "ponto" irão gerar o valor do item **1.2**

**Header:**

```
{
  "alg": "RS256",
  "typ": "JWT"
}
```

**Payload:**

```
{
  "aud": "https://proxy.api.prebanco.com.br/auth/server/v1.1/token",
  "sub": "suaclientkey",
  "iat": 1702580245,
  "exp": 1702583845,
  "jti": 1702580245000,
  "ver": "1.1"
}
```

## 1.2 JWT formado após a conversão em base64 e concatenação com o "."

```
$ eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdwQiOiJodHRwczovL3Byb3h5LmFwaS5wcmViYW5jb20uYnIvYXV0aC9zZXJ2ZXIvdjEuMS90b2t1biIsInN1YiI6ImQ3NTNmMTczLWUxNWEtNDU1Ny05YThmLU0MWU4NzcyOWIwOCIsIm1hdCI6MTcwMjU4MDI0NSwiZXhwIjoxNzAyNTgzODQ1LCJqdGkiOiJlMzMDI1ODAyNDUwMDAsInZ1ciI6IjEuMSJ9
```

**Continuando, para montar a sua assinatura, os passos abaixo deverão ser seguidos:**

Para a assinatura ser montada é necessário usar a chave privada. Para isso podemos usar a ferramenta OpenSSL. Utilize o comando abaixo para obter o JWS.

```
echo -n "${cat jwt.txt}" | openssl dgst -sha256 -keyform pem -sign suporte.teste.com.key.pem | base64 | tr -d '[:space:]' | tr '+/' '-_'
```

**Esse comando irá gerar a string de assinatura, conforme abaixo:**

```
ujK6zn7Jpx1kdon0aVuerKRT9YOEH_Kp6kEI0Frtn7TpQRatqbBB-Pa39CMRFtrY0lIySD3N_NF4CmDi8wgAYS  
AVFeI-V-0V0jkdUWN9VrvJweuRb8yaKzpwYEvV_wbiI_z29jcCrj7N4RleLrFJCi4j15KfTnt_ac6n7ke94Cdk  
98L793DyN4TaVbJq14Z1MvOW9PudBCDt3XT2DiSy6_ug-moUjkjz0FRPcgiJP4SQFYwrH1-_G8XgpfN06fYYZ  
uHPLCZknXB1iozS_HmnqGtyNTbTH1Pr_bj0oS61UZmPcdTy56Qadb98LpbeJg41I81-7r6jP_D6Rcq6zyYmA
```

Agora basta colar essa parte final no seu JWT para que ele seja um JWS.

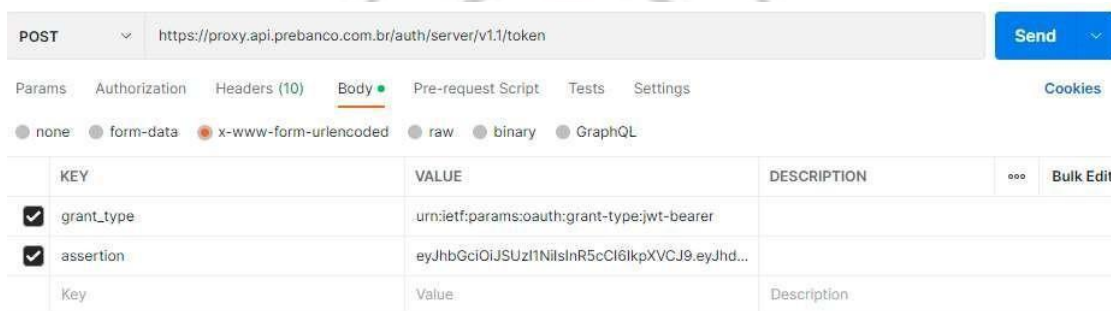
```
$ eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdwQiOiJodHRwczovL3Byb3h5LmFwaS5wcmViYW5jb20uYnIvYXV0aC9zZXJ2ZXIvdjEuMS90b2t1biIsInN1YiI6ImQ3NTNmMTczLWUxNWEtNDU1Ny05YThmLU0MWU4NzcyOWIwOCIsIm1hdCI6MTcwMjU4MDI0NSwiZXhwIjoxNzAyNTgzODQ1LCJqdGkiOiJlMzMDI1ODAyNDUwMDAsInZ1ciI6IjEuMSJ9.ujK6zn7Jpx1kdon0aVuerKRT9YOEH_Kp6kEI0Frtn7TpQRatqbBB-Pa39CMRFtrY0lIySD3N_NF4CmDi8wgAYS  
AVFeI-V-0V0jkdUWN9VrvJweuRb8yaKzpwYEvV_wbiI_z29jcCrj7N4RleLrFJCi4j15KfTnt_ac6n7ke94Cdk98L793DyN4TaVbJq14Z1MvOW9PudBCDt3XT2DiSy6_ug-moUjkjz0FRPcgiJP4SQFYwrH1-_G8XgpfN06fYYZuHPLCZknXB1iozS_HmnqGtyNTbTH1Pr_bj0oS61UZmPcdTy56Qadb98LpbeJg41I81-7r6jP_D6Rcq6zyYmA
```

## 1.3 A chamada do Token:

Após criar o JWS, será possível solicitar um **token JWT (Bearer Token)** para acessar as APIs, esse token ele possui duração de 1Hr do momento da sua geração, e deverá ser utilizado nas chamadas enquanto for válido.

A requisição para chamada de um Bearer Token deverá conter os seguintes itens:

- **Método:** POST
- **URL:** <https://<endereço do ambiente>/auth/server/v1.1/token> (No exemplo abaixo temos o endereço do ambiente de Homologação)
- **Headers**
  - **Content-Type:** application/x-www-form-urlencoded
- **Body**
  - **grant\_type** : urn:ietf:params:oauth:grant-type:jwt-bearer
  - **assertion**: <JWS gerado>



The screenshot shows a REST client interface with the following configuration:

- Method:** POST
- URL:** <https://proxy.api.prebanco.com.br/auth/server/v1.1/token>
- Body Type:** x-www-form-urlencoded
- Body Parameters:**

KEY	VALUE	DESCRIPTION
grant_type	urn:ietf:params:oauth:grant-type:jwt-bearer	
assertion	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhd...	
Key	Value	Description



Ao enviar a chamada, será obtido um resultado similar ao presente abaixo:

[illegible]

Lembrando que, o **bearer token** possui validade de uma hora. É crucial que o mesmo **bearer token** gerado seja reutilizado em todas as requisições nos **endpoints** de serviço durante este período. Essa reutilização evita que toda chamada de negócio necessite ser gerado um novo **token**, consequentemente, evitará possíveis bloqueios de chamada por parte do sistema de segurança do banco.

## CONSUMO DE UMA API APÓS GERAÇÃO DO BEARER TOKEN

### 1. Consumo em homologação

Após todo o processo de criação e geração de um **token JWT (Bearer Token)**, agora iremos demonstrar como consumir um endpoint de qualquer OPEN API server-to-server do banco.

Assim como é feita a assinatura de um **JWT** para envio durante o momento de **autorização**, no momento de consumir APIs é necessário criar uma nova assinatura e enviá-la no **header da requisição**, bem como, adicionar alguns outros campos também durante o processo.

Os campos necessários para chamar as nossas APIs **são padrões para todas as OPEN APIs** do banco, ou seja, independentemente da API que você esteja requisitando, **os campos abaixo serão sempre obrigatórios**, alguns outros campos podem existir, porém, nesse caso, a mudança estará especificada no manual do produto que o cliente for consumir:

<input checked="" type="checkbox"/>	Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxMiJ9.ew0...
<input checked="" type="checkbox"/>	X-Brad-Nonce	1672145026615
<input checked="" type="checkbox"/>	X-Brad-Signature	qpLIWTBcrILZDOL+3MXwZf5FgHcYpZRz4DMBbYo5i3IX...
<input checked="" type="checkbox"/>	X-Brad-Timestamp	2022-12-27T09:43:46-00:00
<input checked="" type="checkbox"/>	X-Brad-Algorithm	SHA256
<input checked="" type="checkbox"/>	access-token	d753f173-e15a-4557-9a8f-e41e87729b08

Logo abaixo consta a explicação de cada um deles e como obter o valor dos mesmos:

O campo **"Authorization"** é um campo que recebe o **token JWT (Bearer Token)** que foi gerado nos passos anteriores, ele deve vir acompanhado da palavra Bearer, assim como está na imagem acima.

O campo **"X-Brad-Nonce"** deverá ser um valor aleatório usado a cada chamada. Podemos usar, por exemplo, a data atual em milissegundos (a data atual em milissegundos, 1672145026618). **OBS: Para fins de facilidade, sugerimos a utilização o mesmo dado do campo "JTI" que você gerou o seu bearer token.**

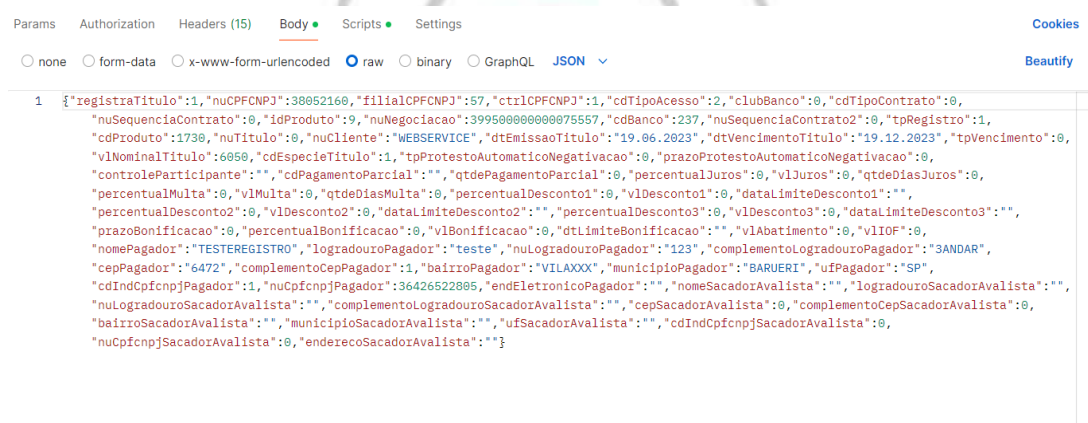
Já o **header "X-Brad-Signature"** é um campo obrigatório para todas as chamadas de consumo de nossas APIs. O seu valor é um JWS que deve ser assinado a partir da

montagem de um "arquivo" .txt ou uma string única com a seguinte estrutura e campos abaixo:

- Na primeira linha do arquivo será inserido o **Método HTTP usado na requisição do endpoint a ser requisitado**. Exemplo, se você for realizar um "POST" para registro de um boleto, a informação presente na primeira linha, será o método POST.
- Na segunda linha, será recebido o **Endpoint** da sua chamada
- Na terceira linha do arquivo, serão os **Parâmetros** da requisição caso eles existam para aquele determinado endpoint. **Caso não exista parâmetros, a linha deverá ficar em branco e será pulada utilizando a quebra de linha "\n".**
- **Na sua quarta linha, será inserido o Body** da requisição, caso não exista um body para a sua chamada, ela deverá seguir o mesmo modelo do parâmetro, ou seja, deverá ficar em branco e **será pulada utilizando a quebra de linha "\n".**

**Ponto importante:** O body da sua requisição deve ser sempre passado em uma única linha no formato de string, sem espaços entre os valores existentes no JSON, essa recomendação serve tanto para o request.txt quanto para a sua requisição propriamente dita.

### Imagem exemplificativa do ponto acima:



- **Na sua quinta linha de montagem, será inserido o Bearer Token** de acesso gerado anteriormente com validade de 1hr.
- **Na sexta linha, o valor será o Nonce** (valor numérico de no máximo dezoito dígitos, **o mesmo usado no header X-Brad-Nonce**)
- **Na sétima linha do arquivo, será inserido o Timestamp**, representando o momento da chamada (o mesmo usado no **header X-Brad-Timestamp**).
- Na oitava e última linha, trará o Algoritmo usado para assinar o JWT, no campo **header "X-Brad-Algorithm"**, **que será o valor: SHA256**

**IMPORTANTE, todas as Quebras de linha, devem ser em formato line feed \n (UNIX)**

Todas essas informações devem ser escritas em um arquivo texto simples, uma linha abaixo da outra, com o arquivo sendo assinado ao final do preenchimento desses dados. A *string* gerada nessa assinatura deverá ser usada no campo ***X-Brad-Signature***.

```
1 POST
2 /v1/boleto-hibrido/registrar-boleto
3
4 {"ctituloCobrCdent":"00026383259","tipoAcesso":"2","cpssoaJuridContr":"2269651","ctpoContrNegoc":
5 eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxMiJ9.ew0KICJ2ZXIiOiAiMS4wIiwNCiAiaXNzIjogImh0dHA6Ly8xMC4xOTQuNjYu
6 1721910301000
7 2024-07-25T16:18:00-00:00
8 SHA256
```

*Exemplo de como o arquivo de texto deve ser preenchido, utilizamos o endpoint da cobrança-híbrida como exemplo, porém, poderia ser qualquer outra API no caso, a diferença seria as particularidades de cada endpoint, como a URL, o body ou os parâmetros.*

```
echo -n "$(cat request.txt)" | openssl dgst -sha256 -keyform pem -sign suporte.teste.com.key.pem |
base64 | tr -d '[:space:]' | tr '+/' '-_'
```

*O Comando usado para assinar o arquivo montado anteriormente, é o mesmo comando utilizado no processo da montagem do seu JWS, acima temos um exemplo.*



Após usar o comando citado anteriormente, será gerado o valor do **"X-Brad-Signature"**.

O **parâmetro chamado "access-token"** presente no header das nossas requisições. O "access-token" possui como valor o seu client Id (client Key), enviado por nós do suporteAPI. Lembrem-se de não confundir com o Bearer Token que possui validade de 1hr, **são dados diferentes**.

***Já o parâmetro "X-Brad-Timestamp" no header, ele irá conter a data e o*** horário de quando o *endpoint* está sendo chamado. Necessário seguir essas regras:

- Formato **"AAAA-MM-DDThh:mm:ss-00:00"**, sendo:
  - **AAAA** = ano com quatro caracteres; **exemplo** "2023"
  - **MM** = mês com dois caracteres; **exemplo** "02" referindo-se ao mês de fevereiro;
  - **DD** = dia com dois caracteres; **exemplo** "15"
  - **T** = parâmetro fixo;
  - **hh** = hora com dois caracteres; **exemplo** "11", referindo-se às 11 da manhã
  - **mm** = minutos com dois caracteres; **exemplo** "38"
  - **ss** = segundos com dois caracteres, **exemplo** "00";
  - -00:00 = diferença para o fuso horário UTC 0:00.
  - Pode ser utilizado UTC -3:00 (fuso horário Brasília)

Sugerimos o uso do seguinte formato abaixo, ignorando os segundos e o fuso horário da sua localidade:

<input checked="" type="checkbox"/>	X-Brad-Timestamp	2024-10-24T11:25:00-00:00
-------------------------------------	------------------	---------------------------

**Após concluir os preenchimentos dos parâmetros presentes nos nossos headers, a requisição ficará da seguinte maneira:**

POST

https://proxy.api.prebanco.com.br/ENDPOINT A SER UTILIZADO

Params

Authorization

Headers (15)

Body

Scripts

Settings

Headers

9 hidden

	Key	Value
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJSUzUxMiJ9.ew0KInZlcil6lClxLjAiLA...
<input checked="" type="checkbox"/>	X-Brad-Signature	EonqjsvZW9GvWzLsfYURCLrnZDEDagrDedsb81BPCpA...
<input checked="" type="checkbox"/>	X-Brad-Nonce	1721910301000
<input checked="" type="checkbox"/>	X-Brad-Timestamp	2024-10-24T11:25:00-00:00
<input checked="" type="checkbox"/>	X-Brad-Algorithm	SHA256
<input checked="" type="checkbox"/>	access-token	d753f173-e15a-4557-9a8f-e41e87729b08

Caso você efetue uma chamada bem sucedida no seu endpoint de negócio, o retorno serão os status 200 OK ou 201 OK

Realizado o teste acima, você estará pronto para integrar em outras **OPEN APIs** do banco.

## 2. Consumo em Produção

---

Após ter concluído todos os testes com sucesso em ambiente de homologação, o próximo passo é solicitar as credenciais do ambiente de produção. Para isso, será necessário adquirir um certificado emitido por Autoridade certificadora, como: *Digicert, CertiSign, Serasa* entre outras. Geralmente o certificado emitido pelas Autoridades Certificadoras é disponibilizado em um arquivo formato “.pfx”.

Este arquivo contém uma “chave pública e uma “chave privada”, ao qual a chave-pública deverá ser extraída do arquivo “.pfx” para envio ao Bradesco. Solicite a empresa emissora do certificado para que lhe forneça os arquivos de chave pública e privada do mesmo.

O Certificado SSL deve ter **padrão ICP-Brasil e ser do Tipo A1** seguindo as características mencionadas anteriormente aqui no manual. Também recomendamos que no momento da aquisição do certificado SSL seja solicitado à empresa que irá gerar o novo certificado que forneça os arquivos de **chave pública** e **privada** no formato “.pem” e “.cer”, facilitando posteriormente o envio do arquivo “.pem” ou “.cer” referente a chave pública para o Bradesco.

**IMPORTANTE:** O consumo das nossas APIs em produção ocorrem na URL <https://openapi.bradesco.com.br>, nunca na <https://proxy.api.prebanco.com.br>

**IMPORTANTE:** No ambiente produtivo só serão aceitos certificados ASSINADOS por entidade certificadora. **Não serão aceitos certificados AUTOASSINADOS.**

## TEMPLATES PARA CONTATO COM O SUPORTEAPI

### 1. Cadastro de *Client ID*

Para cadastro de credenciais e certificado de novos parceiros em ambiente de *homologação* e produção, use o *template* abaixo:

- No campo assunto do e-mail deverá ser preenchido:
  - CAD-HML ou CAD-PRD | NOME DA API | RAZAO SOCIAL | CNPJ
  - **Exemplo:** CAD-HML| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000.00
- Anexar o certificado (**chave pública**) seguindo as regras abaixo:
  - **Formato** .cer, .crt ou. Pem
  - **Validade** mínima de 4 meses e máxima de 3 anos
  - Para ambiente de **homologação somente serão aceitos certificados auto assinados**
- Enviar no conteúdo do e-mail as seguintes informações:
  - **Breve descrição/uso** da aplicação consumidora;
  - **Produto (API) ou serviço** que será consumido;
  - Dois e-mails de **contatos de referência** para renovações de certificados e avisos;

### 2. Renovações

Para renovações de certificados vencidos ou prestes a vencer em ambiente de Homologação ou Produção, seguir o padrão abaixo:

- No campo assunto do e-mail deverá ser preenchido:
  - RNV-HML ou RNV-PRD | NOME DA API | RAZAO SOCIAL | CNPJ
  - **Exemplo:** RNV-HML| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000.00
- Anexar o certificado (**chave pública**) seguindo as regras abaixo:

- **Formato** .cer, .crt ou. Pem
- **Validade** mínima de 4 meses e máxima de 3 anos
- Para ambiente de **homologação somente serão aceitos certificados autoassinados**
- Enviar no conteúdo do e-mail as seguintes informações:
  - **Data** desejada para renovação em produção;
  - **Nome, telefone e e-mail** do responsável por fazer a **validação após renovação** em ambiente produtivo. Em caso de não ocorrer o acompanhamento, por favor, justificar o motivo;
  - **Nome, telefone e e-mail do** responsável pelo certificado. Esse contato será acionado em caso de vencimento e/ou problemas com o certificado;
  - ClientKey do cadastro que será renovado;

### 3. Solicitação de Suporte Técnico e dúvidas

Em caso de dúvidas para conectividade com nossas APIs, entre em contato pelo e-mail: [suporte.api@bradesco.com.br](mailto:suporte.api@bradesco.com.br). Para acionamento do suporte favor enviar o e-mail no padrão abaixo:

#### 3.1 Para dúvidas em homologação sobre a jornada da API:

- O **assunto** do e-mail deve seguir o padrão:
  - DUV-HML| NOME DA API | RAZAO SOCIAL | CNPJ
  - **Exemplo:** DUV-HML| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000-00
- Contextualizar no corpo do e-mail a dúvida com o máximo de **informações** possíveis:
  - **Ambiente** (Homologação)
  - **URL** utilizada pelo cliente para acessar o serviço
  - **Data e hora** da requisição
  - **Body** do request

- **Body** do response

### 3.2 Para dúvidas técnicas de conectividade de API em produção:

- O assunto do e-mail deve seguir o padrão
  - DUV-PRD| NOME DA API | RAZAO SOCIAL | CNPJ
  - **Exemplo:** DUV-PRD| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000-00
- Anexar o **cURL** da chamada executada
- Anexar o **response** contendo o erro
- Contextualizar no corpo do e-mail a dúvida com o máximo de **informações** possíveis



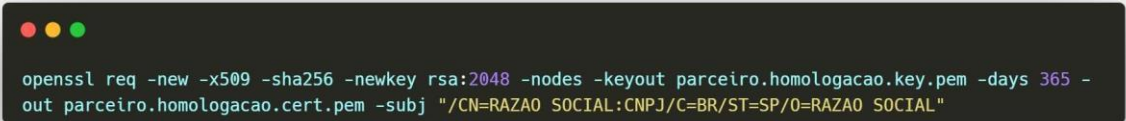
## GUIA DE REFERÊNCIA

### 1- Criando um certificado auto assinado

Para criar certificados auto assinados, a ferramenta que costuma ser a mais usada se chama OpenSSL. Em ambientes Linux ela costuma vir instalada como padrão. Ao executar os comandos são gerados dois arquivos, um sendo a chave pública (cert.pem) e o outro a chave privada (key.pem).

Vale ressaltar que a chave privada **nunca deve** ser compartilhada, pois é ela que garante a autenticidade da chave pública, que será compartilhada conosco.

É necessário que o campo CN do certificado contenha a razão social e CNPJ da empresa, pois essas informações serão usadas para fins de validação interna.



```
openssl req -new -x509 -sha256 -newkey rsa:2048 -nodes -keyout parceiro.homologacao.key.pem -days 365 -out parceiro.homologacao.cert.pem -subj "/CN=RAZAO SOCIAL:CNPJ/C=BR/ST=SP/O=RAZAO SOCIAL"
```

Na imagem acima temos um exemplo de como o comando deve ser executado. Onde o termo "parceiro" aparece, você pode substituir para um nome de sua preferência para fins de organização.

Quando o comando for executado, serão gerados dois arquivos:

- Chave privada (parceiro.homologacao.key.pem)
- Chave pública (parceiro.homologacao.cert.pem)

A chave pública deverá ser compartilhada com o Bradesco e a chave privada deverá ficar sob sua tutela e não ser compartilhada com ninguém. As validações de autoridade serão feitas através da comparação dessas chaves.

## GLOSSÁRIO

### **Open API**

Interfaces de programação de aplicativos abertas e disponíveis publicamente para acesso e integração.

### **Protocolo mTLS**

Autenticação de Transporte de Camada Dupla (mutual TLS), onde tanto o cliente quanto o servidor autenticam uns aos outros com base em seus certificados digitais.

### **Certificado digital**

Arquivo eletrônico que contém informações de identificação e autenticação para um usuário, dispositivo ou aplicativo.

### **OAuth 2.0**

Protocolo de autorização para aplicativos de terceiros que permite o acesso seguro a recursos protegidos de um serviço ou aplicativo.

### **Autenticação**

Processo de verificação de identidade de um usuário ou sistema antes de conceder acesso a recursos protegidos.

### **Autorização**

Processo de permitir ou negar o acesso de um usuário ou sistema a recursos específicos após a autenticação.

### **Client ID**

Identificador exclusivo atribuído a um aplicativo cliente registrado em um sistema de autenticação e autorização.

### **Client Key**

Chave criptográfica privada usada para autenticar um aplicativo cliente em um sistema de autenticação e autorização.

### **Token**

String de caracteres que representa a autorização concedida a um usuário ou aplicativo para acessar recursos protegidos.

### **JWT**

JSON Web Token, um padrão aberto para transmitir informações seguras como tokens de autenticação ou autorização em um formato JSON.

### **Base64**

Método de codificação para representar dados binários em ASCII para transmissão em sistemas que não aceitam dados binários diretamente.

### **Header**

Parte da mensagem HTTP que contém informações sobre a requisição ou resposta.



**Requisição**

Mensagem enviada por um cliente para solicitar uma ação a ser executada por um servidor.

**Endpoint**

URL que um cliente pode acessar para interagir com um serviço ou aplicativo.

**Chave pública e privada**

Par de chaves criptográficas usadas para autenticar usuários ou dispositivos e criptografar dados.

**JWS**

JSON Web Signature, um formato para assinar digitalmente informações JSON.

**Access Token**

Token de autorização usado para acessar recursos protegidos em um serviço ou aplicativo.

**Body**

Parte da mensagem HTTP que contém os dados transmitidos na requisição ou resposta.

**Header**

Parte da mensagem HTTP que contém informações sobre a requisição ou resposta.

**URL**

Endereço da Web que identifica um recurso específico que pode ser acessado por um cliente.

**Payload**

Dados transmitidos na parte do corpo de uma mensagem.

**Query Parameters**

Informações adicionais transmitidas na parte da URL de uma requisição HTTP que podem ser usadas para filtrar ou modificar os resultados retornados.

