Manual de Integração com VIDaaS - Certificado em Nuvem

v1.2

URIs base do Valid PSC

Produção: https://certificado.vidaas.com.br

Homologação: https://hml-certificado.vidaas.com.br

Demonstração: https://demo-certificado.vidaas.com.br

Passo 1 - Cadastro de Aplicação sem Certificado

Para utilizar os serviços de autorização e assinatura é obrigatório cadastrar sua aplicação junto ao Valid PSC, este serviço para cadastro é realizado uma única vez. Abaixo a descrição do serviço para cadastro de aplicação.

Solicitação de cadastro:

- Path : <URI-base>/v0/oauth/application
- Método HTTPS: POST
- Cabeçalho:
  - Content-type: application/json ;
  - Accept: application/json ;
- Parâmetros: formato "application/json;charset=UTF-8":
  - name: obrigatório, nome/descrição da aplicação;
  - comments: obrigatório, observações gerais de uso da aplicação;
  - redirect_uris: obrigatório, URI's autorizadas para redirecionamento (para serviços de código de autorização).
  - email: obrigatório, e-mail para suporte em caso de indisponibilidade, mudança de versão, entre outros.

Exemplo Request:

```
{
    "name": "App Demo",
    "comments": "App Demo",
    "redirect_uris": [https://valid.com.br/],
    "email": "demo@valid.com"
}
```

Exemplo Response:

```
{
   "status": "success",
   "message": "New Client Application registered with Sucess",
   "client_id": "4c9fb552-0387-4e5f-8727-6676fa88dce1",
   "client_secret": "Ny2n3hq67gQEFvH7"
}
```

Observação:

Os parâmetros de retorno **“client_id”** e **“client_secret”** devem ser armazenadas na sua aplicação que serão parâmetros nas chamadas de outros serviços para autorização e assinatura.

Não incluir no cadastro de “`redirect_uris`“ o “push://” pois é gerado automaticamente.

Passo 2 - Serviço para encontrar um titular mediante informação de CPF ou CNPJ

Antes de chamar a aplicação para autenticação, você pode verificar se um CPF ou CNPJ existe no Valid PSC e listar os certificados disponíveis para ele.

Solicitação do serviço:

- Path: <URI-base>/v0/oauth/user-discovery;
- Método HTTPS: POST;
- Parâmetros da requisição: formato "application/json;charset=UTF-8":
  - client_id: obrigatório, deve conter a identificação da aplicação;
  - client_secret: obrigatório, deve conter o segredo associado à aplicação;
  - user_cpf_cnpj: obrigatório, deve conter “CPF” para pessoa física ou “CNPJ” pessoa jurídica;
  - val_cpf_cnpj: obrigatório, deve conter o valor do cpf ou cnpj ;

Exemplo Request:

```
{
    "client_id": "4c9fb552-0387-4e5f-8727-6676fa88dce1",
    "client_secret": "Ny2n3hq67gQEFvH7",
    "user_cpf_cnpj": "CPF",
    "val_cpf_cnpj": "12345678901"
}
```

Exemplo Response:

2.1 - Status S

```
{
   "status": "S",
   "slots": [
       {
            "slot_alias": "a4c2b5bc-ee20-492a-9908-
45d892f2e808",
            "label": "e-CPF A3 em nuvem gold"
       },
       {
            "slot_alias": "8e8353d5-e786-4822-9666-
603bd966ee58",
            "label": "e-CPF A3 em nuvem gold"
       }
    ]
}
```


2.2 - Status N


```
{
   "status": "N"
}
```


Observação:
Status de retorno indicando “S” para resultado positivo ou “N” caso contrário;

Passo 3 - Autorização e Autenticação

Serviço para obter do titular autorização de uso da sua chave privada, com solicitação de fatores de autenticação.

AUTHORIZATION

Solicitação de Autorização:

- Path : <URI-base>/v0/oauth/authorize;
- Método HTTPS: GET;
- Parâmetros da requisição: formato "application/x-www-form-urlencoded"
    - client_id: obrigatório, deve conter a identificação da aplicação;
    - code_challenge: obrigatório, ver RFC 7636;
    - code_challenge_method: obrigatório, valor “S256” (ver RFC 7636);
    - response_type: obrigatório, valor "code";
    - scope: opcional, se não informado, será considerado "single_signature". Possíveis valores para o parâmetro:
        - single_signature: token que permite a assinatura de apenas um conteúdo (hash), sendo invalidado após a sua utilização;
        - multi_signature: token que permite a assinatura de multiplos hashes em uma única requisição, sendo invalidado após a sua utilização;
        - signature_session: token de sessão OAuth que permite várias assinaturas em várias chamadas a API, desde que o token esteja dentro do prazo de validade ou que não tenha sido revogado pela aplicação ou pelo usuário.
    - login_hint: opcional, valor de CPF ou CNPJ a ser informado como filtro para seleção do certificado a ser utilizado.
    - lifetime: opcional, indica o tempo de vida desejado para o token a ser gerado. Inteiro, em segundos;

- redirect_uri: opcional, deve ser igual ou conter uma url informado no Serviço Cadastro de Aplicação. Para solicitar uma notificação deve-se informar apenas o valor “push://” no parâmetro.

Exemplo Request:

```
PATH_BASE/oauth/authorize?client_id=4c9fb552-0387-4e5f-8727-
6676fa88dce1&code_challenge=E9Melhoa2OwvFrEMTJguCHaoeK1t8URWbuGJSstw-
cM&code_challenge_method=S256&response_type=code&scope=signature_session
&login_hint=12345678901&lifetime=900&redirect_uri=http://valid.com.br
```

Exemplo Request com PUSH:

```
PATH_BASE/oauth/authorize?client_id=4c9fb552-0387-4e5f-8727-
6676fa88dce1&code_challenge=E9Melhoa2OwvFrEMTJguCHaoeK1t8URWbuGJSstw-
cM&code_challenge_method=S256&response_type=code&scope=signature_session
&login_hint=12345678901&lifetime=900&redirect_uri=push://
```

Exemplo Response:

```
<REDIRECT_URI>?code=2b15b0e1-bbf2-4e55-99b8-93cf824576b1&state=NONE
```

Exemplo Response com PUSH:

```
code=d402d71c-0918-43ca-a07d-62597f559497
```

Observação:
Para o modo "PUSH NOTIFICATION" deverá realizar o Passo **AUTHENTICATIONS** antes do Passo 4 - Token.

Para autorização via QRCODE
Após solicitação de autorização, redireciona para o titular autorizar o uso

Página de Autorização para o titular da chave privada
O Valid PSC irá retornar com HTTP 302, solicitando redirecionamento para a página de autenticação OAuth para que o titular da chave privada possa fazer autenticação via QRCODE. A página com o QRCODE ficará aguardando por 2 minutos o titular da chave privada autenticar com o aplicativo ViDaas.

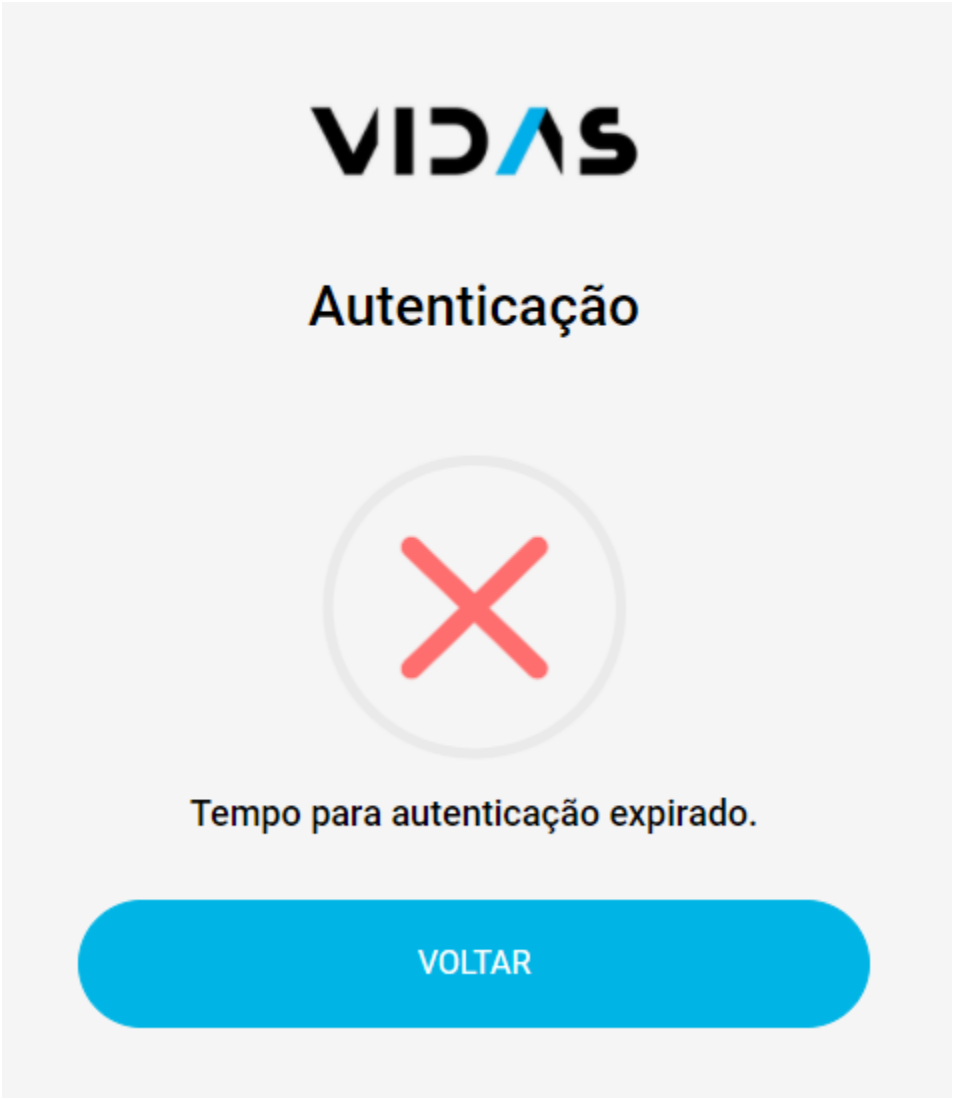
Autenticação pelo titular da chave privada
O titular da chave privada com uso do aplicativo ViDaas efetua a leitura do QRCODE autorizando a aplicação cliente na versão WEB o uso do certificado digital.

Caso o titular possua mais de um certificado, será apresentado os certificados para que o titular faça a escolha do certificado que será autorizado o uso.

Resposta da Requisição de Código de Autorização (Receive Code)
Ao autorizar o uso pelo titular, o Valid PSC emite um código de autorização com tempo de validade curto e retorna para aplicação cliente com uma URI de redirecionamento contendo parâmetros no componente http query, usando o formato "application/x-www- form-urlencoded":

- code: obrigatório, código de autorização gerado pelo Valid PSC, a ser usado na solicitação do token de acesso;
- state: obrigatório caso tenha sido informado na requisição, deverá conter o mesmo valor.

Caso o titular não autorize o uso, o Valid PSC aguardará o tempo de expiração do QRCODE e irá exibir a mensagem conforme imagem abaixo:



AUTHENTICATIONS

Solicitação de Autenticação:

Path : <URI-base>/valid/api/v1/trusted-services/authentications
Método HTTPS: GET
Parâmetros da requisição: formato "application/x-www-form-urlencoded"
code: obrigatório, deve conter código retornado do Serviço de Solicitação de Autorização (**Authorization**);

Exemplo de Solicitação de Autenticação

```
<URI-base>/valid/api/v1/trusted-services/authentications?code=d402d71c-0918-43ca-a07d-62597f559497
```

Exemplo de Resposta

```
{
    "authorizationToken":
"eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiZGlyIn0..
```

```
nYWhIcwNUH_22Upe1BSUTQ.
oXT7UF2Mvtm5C6CjpdEGxcL_9XM86oNh4w0iGgUkQVGBla0CNnNW0_QbGx73Ldnu81kydOuz
tSj3wfWUQf3t7IftvVMuyfdi-
gW4_lz1LcC2q3p9N32iSEGb5VPzzSKqiZGa3asfMgEPjr3xYo7Lo3biTtbVPrChPLHslMi--
b7DXXOIZ23N2R5bCT2_h6pj6PyBnXsEWl5uaF9v5PSXsQ.ZuLdlRZkfGBoqrxbj5tgTg",
    "redirectUrl": "push://<URI gerada no cadastro de aplicação>?
code=8b1bde77-3647-4d76-1289-a2ec97c75a4d&state=NONE"
}
```


Passo 4 - Token

- Path : <URI-base>/v0/oauth/token;
- Método HTTPS: POST;
- Parâmetros da requisição: formato "application/x-www-form-urlencoded"
    - grant_type: obrigatório, valor "authorization_code";
    - client_id: obrigatório, deve conter a identificação da aplicação;
    - client_secret: obrigatório, deve conter o segredo associado à aplicação;
    - code: obrigatório, deve conter código de autorização retornado do Serviço Código de Autorização;
    - redirect_uri: opcional, deve ser igual ao informado no Serviço Código de Autorização;
    - code_verifier: obrigatório, correspondendo a code_challenge enviado na Requisição de Código de Autorização (ver RFC 7636).

Observação:
Para o método Authorize QRCODE (sem o push://) deve-se alterar o campo "code" pelo "code" retornado na URL.
Para o método Authorize PUSH deve-se alterar o campo "code" pelo "authorizationToken" retornado no Authentications.

Exemplo Request:

```
{
    "grant_type": "authorization_code",
    "client_id": "4c9fb552-0387-4e5f-8727-6676fa88dce1",
    "client_secret": Ny2n3hq67gQEFvH7,
    "code": "<authorization code>",
    "code_verifier": "dBjftJeZ4CVP-mB92K27uhbUJU1p1r_wW1gFWFOEjXk"
}
```

Exemplo Response:


```
{
    "access_token": "eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiZGlyIn0..
2tk9rh8yisesxBm1tNNcUg.z6VZu-HZJk-
a9EDBSAgDrtZWgYn5je__nCc6uOOrl3wsCrzWT5G0SMUHpuX3McdBk0uIJ85cMOe3MFn75Pe
5mfhlmdLtRUtnX_tJmg8rW6dU7mg4nR4XlyMmWYy-
Yep_2dIM2xni0sWUplPxUCLg9dl7_aeVTB_U9TmsXOYCJNMYSJfjPErsthUNHWJHzUIOg-
2Otj9gkq_EBLr0jYVWCw.IPOs5b_o6yKmz2Q24zYYvA",
    "token_type": "Bearer",
    "expires_in": 900,
    "scope": "signature_session",
```

```
    "authorized_identification": "12345678901",
    "authorized_identification_type": "CPF"
}
```

Passo 5 - Signature

- Path: <URI-base>/v0/oauth/signature
- Método HTTPS: POST
- Cabeçalho: Re: Nova Versão da IN PSC
    - Content-type: application/json;
    - Accept : application/json;
    - Authorization: Bearer access_token;
        - Parâmetros: formato "application/json;charset=UTF-8" :
            - certificate_alias: opcional, identificador do certificado correspondente à chave utilizada na assinatura;
            - hashes: conjunto com valores a serem assinados. Cada elemento do conjunto conterá:
                - id: identificador do conteúdo a ser assinado;
                - alias: forma legível do identificador do conteúdo;
                - hash: conteúdo a ser assinado;
                - hash_algorithm: Object Identifier (OID) do algoritimo de hash. Por exemplo, para SHA256 utilize o OID 2.16.840.1.101.3.4.2.1;
                - signature_format: obrigatório:
                    - RAW: resultado direto (em base64) da operação RSA sobre o hash informado na requisição.
                    - CMS detached (PKCS#7), contendo os seguintes atributos assinados:
                      - contentType
                      - signingTime (hora do PSC)
                      - messageDigest (hash informado pela aplicação na requisição)
                      - signingCertificateV2 (certificado do assinante);
                - padding_method: opcional,
                - pdf_signature_page: opcional, página acrescentada no PDF mostrando a assinatura digital, podendo ter o valor "true" ou "alse", por padrão é definido o valor "false"
                - base64_content: opcional, conteúdo em base64 a ser assinado. Por exemplo Texto ou PDF a ser assinado em base64

Observação:
A aplicação suporta os seguintes signature_format:
RAW, CMS, CAdES_AD_RT, CAdES_AD_RB, PAdES_AD_RT, PAdES_AD_RB, RAW_DIGESTED_DATA.

e padding_method:
NONE, PKCS1V1_5, PSS.

Exemplo Request:

```
{
    "hashes": [
          {
              "id": "dummy assinatura",
              "alias": "dummy.pdf",
              "hash": "FqulOTrXLABB9WAK08LFLsQ3ovDH/Aj638PA/pZB16M=",
              "hash_algorithm": "2.16.840.1.101.3.4.2.1",
              "signature_format": "RAW"
          }
    ]
}
```

Exemplo Response:

```
{
    "signatures": [
         {
             "id": "dummy assinatura",
             "raw_signature": "my signature base64"
         }
    ],
    "certificate_alias": "CERTIFICADO TESTE"
}
```