PUBCON - Documentação técnica da API

1. OBJETIVO

Esta documentação descreve os parâmetros técnicos para acessar a api do sistemas de publicações - PUBCON, possibilitando o desenvolvimento de uma solução automatizada para a consulta de dados.

Para acessar a api é necessário o cadastramento do usuário para obtenção do código do usuário e da chave de acesso que deverão ser utilizadas em todas as chamadas da api.

2. RETORNO DA API

O retorno de todas as consultas possuem um conteúdo json com a seguinte estrutura:

SUCESSO

```
{
      “error”: false,
      “message”: “”,
      “data”: { }
}
```

ERRO

```
{
      “error”: true,
      “message”: “mensagem do erro”,
      “data”: null
}
```

Os dados de uma consulta específica, quando houver, serão retornados na chave “data”.

As possíveis mensagens de erros retornadas pela API podem ser:

- Requisição inválida: ocorrerá quando estiver faltando algum parâmetro do cabeçalho HTTP.

- Usuário não cadastrado: ocorrerá quando o usuário ainda não estiver cadastrado no sistema.

- Assinatura da requisição inválida: ocorre quando a assinatura gerada pelo cliente é diferente da assinatura gerada pela API.

- Ocorreu timeout: ocorre quando houver um atraso no recebimento da requisição pela API acima do limite de tempo configurado, que é de 1 minuto. Neste caso é necessário verificar o valor do parâmetro de cabeçalho PUB_API_TIMESTAMP.

- Usuário excedeu o limite de requisições diário: ocorre quando o usuário ultrapassar o limite diário permitido de requisições à api..

3. CABEÇALHO HTTP

Todas as requisições à api, GET ou POST, deverão conter os seguintes parâmetros no cabeçalho:

ContentType = application/json
PUB_API_USER = código do usuário

PUB_API_TIMESTAMP = timestamp, data e hora da criação da requisição em milissegundos UTC
PUB_API_SIGN = assinatura com HMAC SHA256, use a chave de acesso

O código do usuário e a chave de acesso serão enviados por email quando o usuário se cadastrar no sistema.
Para gerar a assinatura concatene o timestamp, código do usuário e objeto enviado no corpo da requisição, se houver, e aplique HMAC SHA256 com a chave de acesso.

Exemplo:

string texto = timestamp + codigoUsuario + obj
string sign = HMACSHA256(texto, chaveAcesso)

4. URL BASE

A URL base para as chamadas da api tem o seguinte formato:

https://servicos.sorocaba.sp.gov.br/pubcon-consulta/cliente/

5. LIMITES

As quantidade de requisições feitas na api serão limitadas por um valor diário, sendo esse reiniciado a cada dia.

6. ROTAS

6.1. Consulta de status da publicação

Retorna uma lista de status da publicação.

URL: /statusPublicacao

Método: GET

Parâmetros: Nenhum

Retorno

Campo	Tipo	Descrição
Id	inteiro	identificador do status da publicação
Nome	texto	descrição dos tipos de status de cada publicação

6.2. Consulta de tipos de parceria

Retorna uma lista de tipos de parceria.

URL: /tipoParceria

Método: GET

Parâmetros: Nenhum

Retorno

Campo	Tipo	Descrição
IdTipoParceria	inteiro	identificador da parceria
IdTipoDocSup	inteiro	identificador do tipo de documento
Descricao	texto	descrição complementar da parceria
Observacao	texto	observação

6.3. Consulta unidade

Retorna uma lista de unidades da administração.

URL: /unidade

Método: GET

Parâmetros: Nenhum

Retorno

Campo	Tipo	Descrição
Id	inteiro	identificador da secretaria
Nome	texto	nome da secretaria responsável
Sigla	texto	sigla da secretaria

6.4. Consulta de publicação

Retorna uma lista de publicações com base em um filtro.

Parâmetros enviados no corpo da requisição no formato json. Somente os campos que tiverem valor devem ser enviados, campos vazios ou nulos devem ser omitidos.

URL: /publicacao

Método: POST

Parâmetros

Campo	Tipo	Descrição
AnoProcesso	inteiro	ano de abertura do processo
NumeroProcesso	inteiro	número do processo

CodigoAssunto	inteiro	código do assunto
Assunto	texto	descrição do assunto resumida do processo
CnpjOrganizacao	texto	cnpj da organização no formato 00.000.000/0000-00
NomeOrganizacao	texto	nome da organização (razão social)
DataPublicacao	texto	data da publicação no formato dd/mm/aaaa
IdUnidade	inteiro	identificador da secretaria responsável pela publicação
IdStatusPublicacao	inteiro	identificador dos tipos de status da publicação 1 = Em Andamento 2 = Encerrado 3 = Cancelado 4 = Suspenso
NumeroParceria	texto	número da parceria
IdTipoParceria	inteiro	identificador dos tipos de parceria 22961 = Convênios 22910 = Termo de Colaboração 22911 = Termo de Fomento 22912 = Acordo de Cooperação 22898 = Convênios Cadastrados 22959 = Contrato de Gestão - Convênios

Retorno

Campo	**Tipo**	**Descrição**
Id	inteiro	identificador da publicação
IdStatusPublicacao	inteiro	identificador do status da publicação
AnoProcessoSiat	inteiro	ano do processo administrativo
NumeroProcessoSiat	inteiro	número do processo administrativo
DigitoVerificadorProcessoSiat	inteiro	dígito verificador do processo administrativo
CodigoAssuntoSiat	inteiro	código do assunto
AssuntoSiat	texto	descrição do assunto de forma resumida
Descricao	texto	descrição complementar da publicação
DataAbertura	texto	data de abertura da publicação
DataPublicacao	texto	data da publicação do processo no portal
IdUnidade	inteiro	identificador da secretaria responsável pela publicação
DataHoraCriacao	texto	data e hora da criação da publicação no portal
DataHoraAlteracao	texto	data e hora da alteração da publicação no portal
StatusPublicacao	objeto	objeto representando o status da publicação
Unidade	objeto	objeto representando a secretaria responsável pelo convênio

6.5. Consulta de convênio

Retorna uma lista de convênios.

Parâmetros enviados na url.

URL: /convenio?numero=<numeroProcesso>&ano=<anoProcesso>

Método: GET

Parâmetros:

- numero: inteiro

- ano: inteiro

Retorno

Campo	Tipo	Descrição
NumeroParceria	texto	número da parceria
ValorInicialParceria	decimal	valor inicial da parceria
ValorAtualParceria	decimal	valor atual da parceria
NumeroProcesso	texto	número do processo administrativo
ProcessoConam	texto	número do processo no Portal da Transparência, constante nos dados dos empenhos
Modalidade	texto	descrição do tipo de modalidade
IdTipoParceria	inteiro	identificador do tipo de parceria
TipoTermo	texto	descrição do tipo de termo
NaturezaConvenio	texto	natureza do convênio
Exercicio	inteiro	exercício da publicação do convênio
NomeOrganizacao	texto	nome da organização (razão social)
NomeFantasia	texto	nome de fantasia da organização
Objeto	texto	descrição resumida do objeto do convênio
DataAssinatura	texto	data de assinatura do convênio
DataFimParceria	texto	data final da parceria
DataInicioParceria	texto	data de início da execução da parceria
CnpjOrganizacao	texto	cnpj da organização
SecretariaResponsavel	texto	nome da secretaria atualmente responsável
TotalEmpenhadoConam	decimal	valor total empenhado para o convênio
TotalProcessadoConam	decimal	valor total processado para o convênio
ValoresLiberados	decimal	valor liberado para a entidade
Saldo	decimal	Saldo do empenho para o convênio

6.6. Consulta de aditivo do convênio

Retorna uma lista de aditivos do convênio.

Parâmetro enviado na url.

URL: /convenioAditivo?nrParceria=<numeroParceria>

Método: GET

Parâmetro:

- nrParceria: texto, formato 000000/0000 (numero/ano)

Retorno

Campo	Tipo	Descrição
IdTermo	inteiro	identificador do termo do aditivo
NumeroParceria	texto	número do convênio sobre o qual se aplica o aditivo
TermoAditivo	texto	número do termo aditivo
TipoTermo	texto	motivo do aditivo
DataInicio	texto	data de concessão do aditivo
Valor	decimal	valor do aditivo
Operacao	texto	identifica a operação registrada, sendo crédito (C) para o efeito aditivo ou débito (D) para uma supressão
Descricao	texto	descrição do aditivo
ValorFmt	texto	valor formatado (moeda)

6.7. Consulta de apostilamento do convênio

Retorna uma lista de apostilamento do convênio.

Parâmetro enviado na url.

URL: /convenioApostilamento?nrParceria=<numeroParceria>

Método: GET

Parâmetro:

- nrParceria: texto, formato 000000/0000 (numero/ano)

Retorno

Campo	Tipo	Descrição
IdTermo	inteiro	identificador do termo de apostilamento
NumeroParceria	texto	número do convênio sobre o qual se aplica o apostilamento

TermoApostilamento	texto	número do termo de apostilamento
TipoTermo	texto	motivo do apostilamento
DataInicio	texto	data inicial do apostilamento
DataFim	textol	data inicial do apostilamento, somente em caso de correção.
Valor	decimal	valor do apostilamento em caso de correção.
Operacao	texto	identifica se o termo trata de correção de valores, registrando débito (D) ou crédito (C)
Descricao	texto	descrição da necessidade do apostilamento
ValorFmt	texto	valor formatado (moeda)

6.8. Consulta de prorrogação do convênio

Retorna uma lista de prorrogação do convênio.

Parâmetro enviado na url.

URL: /convenioProrrogacao?nrParceria=<numeroParceria>

Método: GET

Parâmetros:

- nrParceria: texto, formato 000000/0000 (numero/ano)

Retorno

Campo	**Tipo**	**Descrição**
IdTermo	inteiro	identificador do termo de prorrogação
NumeroParceria	texto	número do convênio objeto do termo de prorrogação
TermoProrrogacao	texto	número da prorrogação
TipoTermo	texto	motivo da prorrogação
DataInicio	texto	data inicial da prorrogação
DataFim	texto	data final da prorrogação
Valor	decimal	valor da prorrogação
Descricao	texto	descrição da prorrogação
ValorFmt	texto	valor formatado (moeda)

6.9. Consulta de prestação de contas

Retorna uma lista de prestação de contas do convênio.

Parâmetro enviado na url.

URL: /convenioPrestacaoConta?nrParceria=<numeroParceria>

Método: GET

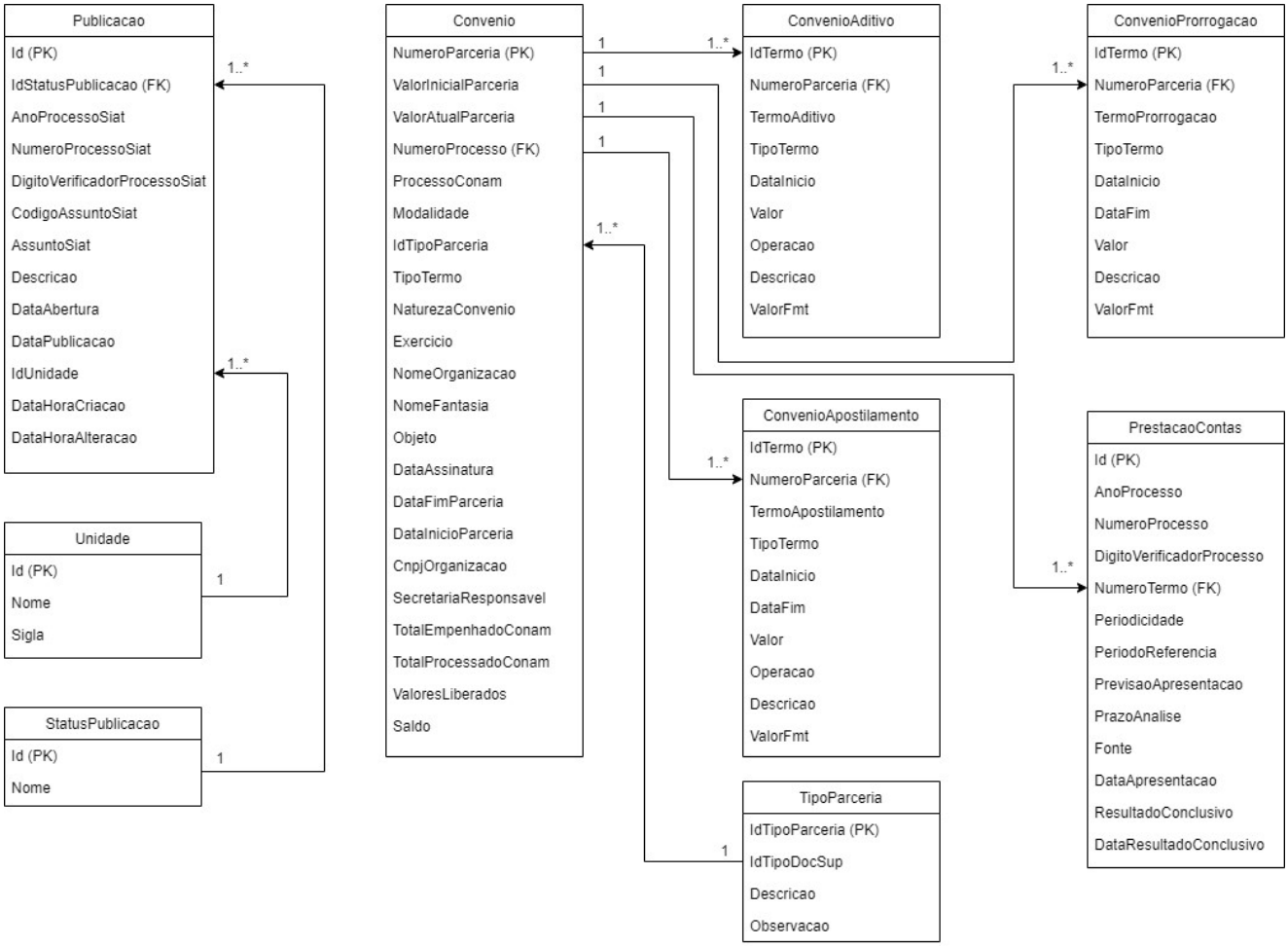
Parâmetro:

- nrParceria: texto, formato 000000/0000 (numero/ano)

Retorno

Campo	Tipo	Descrição
Id	inteiro	identificador da prestação de contas
AnoProcesso	inteiro	ano do processo administrativo
NumeroProcesso	inteiro	número do processo administrativo
DigitoVerificadorProcesso	inteiro	dígito verificador do processo administrativo
NumeroTermo	texto	número do convênio objeto da prestação de contas
Periodicidade	texto	periodicidade definida no convênio para a prestação de contas (mensal, bimestral, trimestral, por exemplo)
PeriodoReferencia	texto	período de referência da prestação de contas, conforme definido no convênio
PrevisaoApresentacao	texto	previsão da data de apresentação da prestação de contas, conforme definido no convênio
PrazoAnalise	texto	prazo para análise da prestação de contas, conforme definido no convênio
Fonte	texto	fonte de recurso para a qual a conta deve ser prestada (municipal, estadual e/ou federal)
DataApresentacao	texto	data da efetiva apresentação da prestação de contas pela entidade conveniada
ResultadoConclusivo	texto	resultado conclusivo sobre a prestação de contas
DataResultadoConclusivo	texto	data de emissão do resultado conclusivo

7. DIAGRAMA DO MODELO DE DADOS



ANEXO - Exemplo de chamadas à API do PUBCON, usando a linguagem C#

```
using Newtonsoft.Json;
using Newtonsoft.Json.Converters;
using PubconApi.Data;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;

namespace PubconApi
{
    public class ApiWeb
    {
        public struct HttpResult
        {
            public Boolean Erro;
            public string MensagemErro;
        }

        public HttpResult httpResult;

        int idUsuario;
        string secret;

        string rtBaseUrl = "";
        string rtPublicacao = "/publicacao";
        string rtConvenio = "/convenio?numero=<paramNumero>&ano=<paramAno>";
        string rtConvenioAditivo = "/convenioAditivo?nrParceria=<paramNrParceria>";
        string rtConvenioApostilamento =
"/convenioApostilamento?nrParceria=<paramNrParceria>";
        string rtConvenioProrrogacao =
"/convenioProrrogacao?nrParceria=<paramNrParceria>";
        string rtPrestacaoConta = "/prestacaoConta?nrParceria=<paramNrParceria>";
        string rtStatus = "/statusPublicacao";
        string rtTipoParceria = "/tipoParceria";
        string rtUnidade = "/unidade";

        public ApiWeb(int idUs, string senhaUs, string url)
        {
            idUsuario = idUs;
            secret = senhaUs;
            if (!string.IsNullOrWhiteSpace(url)) rtBaseUrl = url;
        }


        private string GetSignature(int idUsuario, string secret, string nonce, object obj)
        {
            string texto = nonce + idUsuario.ToString();

            if (obj != null) texto = texto + JsonConvert.SerializeObject(obj);

            return GerarHash(texto, secret);
        }


        private string GetTimeStamp()
        {
            long nonce =
Convert.ToInt64(TimeSpan.FromTicks(DateTime.UtcNow.Ticks).TotalMilliseconds);
            return nonce.ToString();
```

```
}


private string GerarHash(string texto, string secret)
{
    HMACSHA256 hashMaker;
    hashMaker = new HMACSHA256(Encoding.UTF8.GetBytes(secret));

    byte[] data = Encoding.UTF8.GetBytes(texto);
    byte[] hash = hashMaker.ComputeHash(data);
    string signature = GetHexString(hash);

    return signature;
}


private string GetHexString(byte[] bytes)
{
    StringBuilder sb = new StringBuilder(bytes.Length * 2);
    foreach (byte b in bytes)
    {
        sb.Append(String.Format("{0:x2}", b));
    }

    return sb.ToString();
}

private string ValidarRetorno(string response)
{
    ResponseData resp = JsonConvert.DeserializeObject<ResponseData>(response);

    if (!resp.Error)
    {
        return resp.Data.ToString();
    }
    else
    {
        httpResult.Erro = true;
        httpResult.MensagemErro = resp.Message;

        return "";
    }
}


private string SendRequest(string url, object obj, string metodo)
{
    string nonce = GetTimeStamp();

    string signRequest = GetSignature(idUsuario, secret, nonce, obj);

    HttpWebRequest wr = WebRequest.Create(url) as HttpWebRequest;
    wr.ContentType = "application/json";
    wr.Headers.Add("PUB_API_USER", idUsuario.ToString());
    wr.Headers.Add("PUB_API_TIMESTAMP", nonce);
    wr.Headers.Add("PUB_API_SIGN", signRequest);
    wr.Method = metodo;

    string response = null;

    try
    {
        // adiciona obj ao corpo da requisição
        if (obj != null && metodo == "POST")
        {
            using (var streamWriter = new StreamWriter(wr.GetRequestStream()))
```

```
                {
                    string json = JsonConvert.SerializeObject(obj);
                    streamWriter.Write(json);
                }
            }

            HttpWebResponse resp = wr.GetResponse() as HttpWebResponse;
            StreamReader sr = new StreamReader(resp.GetResponseStream());
            response = sr.ReadToEnd();
            sr.Close();

            httpResult.Erro = false;
            httpResult.MensagemErro = "";

            response = ValidarRetorno(response);
        }
        catch (WebException ex)
        {
            httpResult.Erro = true;
            httpResult.MensagemErro = ex.Message;
            response = "";
        }

        return response;
    }


    public List<Publicacao> ConsultarPublicacao(int numeroProcesso, int anoProcesso)
    {
        string url = rtBaseUrl + rtPublicacao;
        string metodo = "POST";

        RequestData obj = new RequestData { NumeroProcesso = numeroProcesso.ToString(),
AnoProcesso = anoProcesso.ToString() };
        string response = SendRequest(url, obj, metodo);

        List<Publicacao> lista = null;

        if (!string.IsNullOrWhiteSpace(response))
        {
            lista = JsonConvert.DeserializeObject<List<Publicacao>>(response);
        }

        return lista;
    }


    public List<Convenio> ConsultarConvenio(int numeroProcesso, int anoProcesso)
    {
        string url = rtBaseUrl + rtConvenio.Replace("<paramNumero>",
numeroProcesso.ToString()).Replace("<paramAno>", anoProcesso.ToString());

        string metodo = "GET";

        string response = SendRequest(url, null, metodo);

        List<Convenio> lista = null;

        if (!string.IsNullOrWhiteSpace(response))
        {
            lista = JsonConvert.DeserializeObject<List<Convenio>>(response);
        }

        return lista;
    }
```

```
        public List<ConvenioAditivo> ConsultarConvenioAditivo(string nrParceria)
        {
            string url = rtBaseUrl + rtConvenioAditivo.Replace("<paramNrParceria>",
nrParceria);

            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<ConvenioAditivo> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<ConvenioAditivo>>(response);
            }

            return lista;
        }


        public List<ConvenioApostilamento> ConsultarConvenioApostilamento(string
nrParceria)
        {
            string url = rtBaseUrl + rtConvenioApostilamento.Replace("<paramNrParceria>",
nrParceria);

            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<ConvenioApostilamento> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista =
JsonConvert.DeserializeObject<List<ConvenioApostilamento>>(response);
            }

            return lista;
        }


        public List<ConvenioProrrogacao> ConsultarConvenioProrrogacao(string nrParceria)
        {
            string url = rtBaseUrl + rtConvenioProrrogacao.Replace("<paramNrParceria>",
nrParceria);

            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<ConvenioProrrogacao> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista =
JsonConvert.DeserializeObject<List<ConvenioProrrogacao>>(response);
            }

            return lista;
        }


        public List<PrestacaoConta> ConsultarPrestacaoConta(string nrParceria)
        {
```

```
            string url = rtBaseUrl + rtPrestacaoConta.Replace("<paramNrParceria>",
nrParceria);

            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<PrestacaoConta> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<PrestacaoConta>>(response);
            }

            return lista;
        }


        public List<StatusPublicacao> ConsultarStatusPublicacao()
        {
            string url = rtBaseUrl + rtStatus;
            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<StatusPublicacao> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<StatusPublicacao>>(response);
            }

            return lista;
        }


        public List<TipoParceria> ConsultarTipoParceria()
        {
            string url = rtBaseUrl + rtTipoParceria;
            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<TipoParceria> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<TipoParceria>>(response);
            }

            return lista;
        }


        public List<Unidade> ConsultarUnidade()
        {
            string url = rtBaseUrl + rtUnidade;
            string metodo = "GET";

            string response = SendRequest(url, null, metodo);

            List<Unidade> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<Unidade>>(response);
```

```
            }

            return lista;
        }

    }
}
```

Use a classe `ApiWeb` assim:

```
int idUs = 1; // seu código de usuário
string codigoAcesso = "123"; // seu código de acesso
string url = "https://servicos.sorocaba.sp.gov.br/pubcon-consulta/cliente"; // url da api

int numero = 1;
int ano = 2018;

ApiWeb api = new ApiWeb(idUs, codigoAcesso, url);

var lista = api.ConsultarPublicacao(numero, ano);

if (!api.httpResult.Erro)
    MessageBox.Show("Consulta realizada com sucesso");
else
    MessageBox.Show(api.httpResult.MensagemErro);
```