PUB - Documentação técnica da API

1. OBJETIVO

Esta documentação descreve os parâmetros técnicos para acessar a api do sistemas de publicações - PUB, 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/pub-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
IdGrupoStatusPublicacao	inteiro	identificador do grupo do status
Nome	texto	descrição dos tipos de status de cada publicação

6.2. Consulta de modalidade

Retorna uma lista de modalidades de publicação.

URL: /modalidade

Método: GET

Parâmetros: Nenhum

Retorno

Campo	Tipo	Descrição
CodigoModalidade	inteiro	código da modalidade
Modalidade	texto	descrição dos tipos de modalidade

6.3. 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
Exercicios	texto	exercício da publicação (caso tenha mais de um ano, separa-los com ponto e vírgula), exemplo 2010;2011
DataAberturaInicial	texto	data de abertura inicial no formato dd/mm/aaaa
DataAberturaFinal	texto	data de abertura final no formato dd/mm/aaaa
CnpjFornecedor	inteiro	cnpj do fornecedor sem máscara
NomeFornecedor	texto	nome do fornecedor (razão social)
DescricaoItem	texto	descrição do item
CodigoModalidade	inteiro	código da modalidade 1 = Concurso 2 = Convite 3 = Tomada de Preços 4 = Concorrência 5 = Dispensa de Licitação 6 = Inexigibilidade 7 = Pregão Eletrônico 8 = Compra Eletrônica 9 = Outros/Não Aplicável 10 = Leilão 11 = Pregão Presencial 13 = Comunicados
AnoProcesso	inteiro	ano de criação do processo
NumeroProcesso	inteiro	número do processo (CPL)

NumeroEdital	inteiro	número do edital
DescricaoObjeto	texto	descrição do objeto
IdStatusPublicacao	inteiro	status da publicação 1 = Em Andamento 2 = Encerrada 3 = Cancelada 4 = Suspensa 5 = Em Execução Contratual 6 = Revogada

Retorno

Campo	Tipo	Descrição
AnoPublicacao	inteiro	ano da publicação
CodigoModalidade	inteiro	código da modalidade
CodigoProcesso	texto	número e ano do processo no formato 000000/0000
DataAbertura	texto	data de abertura do certame
DataHoraAlteracao	texto	data e hora da alteração da publicação
DataHoraCriacao	texto	data e hora da criação da publicação
DataPublicacao	texto	data da publicação
DescricaoObjeto	texto	descrição do objeto do processo
Id	inteiro	identificador da publicação
IdProcessoSim	inteiro	identificador do processo no sistema SIM
IdStatusPublicacaoProcesso	inteiro	identificador do status da publicação
Modalidade	texto	descrição do tipo de modalidade
NumeroEdital	inteiro	número do edital
StatusPublicacaoProcesso	objeto	objeto representando status da publicação

6.4. Consulta de itens do processo

Retorna uma lista de itens do processo com base em um filtro.

Parâmetros enviados na url.

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

Método: GET

Parâmetros:

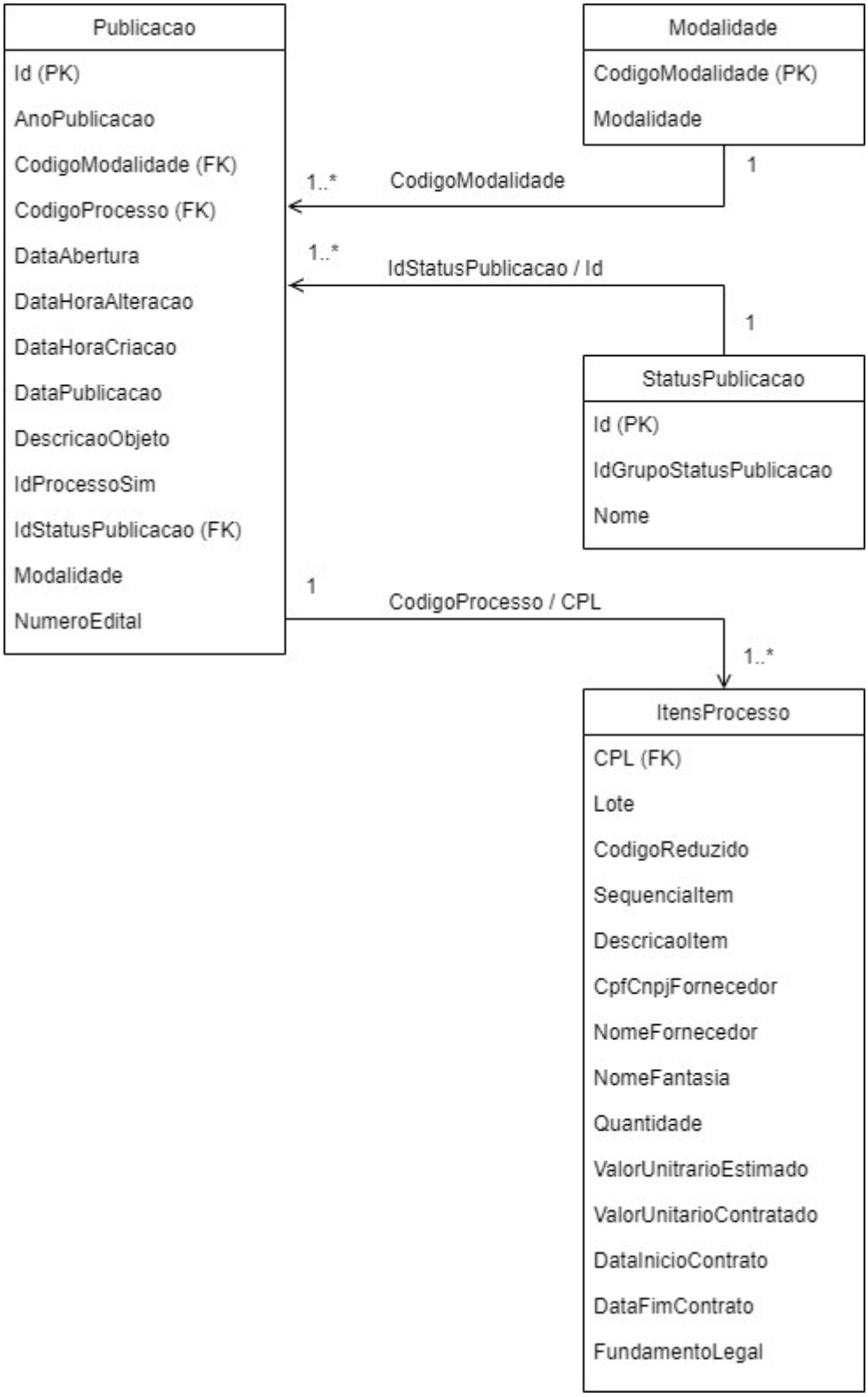
- numero: inteiro

- ano: inteiro

Retorno

Campo	Tipo	Descrição
Cpl	texto	número e ano do processo no formato 000000/0000
Lote	texto	lote do processo
CodigoReduzido	texto	código reduzido do produto
SequencialItem	texto	identificação do item de forma sequencial
DescricaoItem	texto	descrição do item contratado
CpfCnpjFornecedor	texto	cpf ou cnpj do fornecedor
NomeFornecedor	texto	nome do fornecedor (razão social)
NomeFantasia	texto	nome de fantasia do fornecedor
Quantidade	decimal	quantidade contratada
ValorUnitarioEstimado	decimal	valor unitário estimado
ValorUnitarioContratado	decimal	valor unitário contratado
DataInicioContrato	texto	data da ordem de início do contrato
DataFimContrato	texto	data final do contrato
FundamentoLegal	texto	descrição do fundamento legal de acordo com a lei 8666/93

7. DIAGRAMA DO MODELO DE DADOS



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

```
using Newtonsoft.Json;
using PubApi.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 PubApi
{
    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 rtStatus = "/statusPublicacao";
        string rtModallidade = "/modalidade";
        string rtItens = "/itensProcesso?numero=<paramNumero>&ano=<paramAno>";


        public ApiWeb(int idUs, string codigoAcesso, string url)
        {
            idUsuario = idUs;
            secret = codigoAcesso;
            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<PublicacaoProcesso> 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<PublicacaoProcesso> lista = null;

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

            return lista;
        }

        public List<ItemProcesso> ConsultarItens(int numeroProcesso, int anoProcesso)
        {
            string url = rtBaseUrl + rtItens.Replace("<paramNumero>",
numeroProcesso.ToString()).Replace("<paramAno>", anoProcesso.ToString());

            string metodo = "GET";

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

            List<ItemProcesso> lista = null;

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

            return lista;
        }

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

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

            List<StatusPublicacaoProcesso> lista = null;

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

            return lista;
        }

        public List<ModalidadeDto> ConsultarModalidade()
        {
            string url = rtBaseUrl + rtModallidade;
            string metodo = "GET";

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

            List<ModalidadeDto> lista = null;

            if (!string.IsNullOrWhiteSpace(response))
            {
                lista = JsonConvert.DeserializeObject<List<ModalidadeDto>>(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/pub-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);
```