Pulse

Pulse Common Clinical Data API

Implementation Guide

Version 1.0

11/21/2022

VERSION HISTORY

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
1.0	Rob Sprague	11/21/2022			Original Draft

Pulse

Table of Contents

Pulse

1 Introduction

1.1 Who is this document for?

This REST Common Clinical Data API (henceforth referred to as The API) Implementation Guide is for the reference of developers wanting to make a custom integration to the Pulse REST API to retrieve Common Clinical Data.

1.2 System Overview

The API allows 3rd party vendors to build clients to access patient data in Pulse. It uses Authentication Server 3 which implements a Patient Token system to provide a high level of security.

2 Starting Your Implementation

2.1 REST and JSON

REST (REpresentational State Transfer) is a web service architectural style. In general it means that aspects of the HTTP protocol are used to retrieve or manipulate data from a remote system. As with HTTP, a RESTful API is called with the combination of a URL and a verb. The verb indicates the type of action that is requested.

The API only utilizes the POST verb for all operations.

JSON (JavaScript Object Notation) is a notation style used to represent complex object structures in a serialized manner (i.e. transferable over the Internet). It has a similar role to XML but is much less verbose and therefore faster. All request and response messages will be JSON messages.

2.2 Security

The call to PulseAuthenticationtoken will return a token, and it will be used for all subsequent data requests for that session.

SSL is used for transport security. The SSL protocol runs above TCP/IP and below higher-level protocols such as HTTP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to and SSL-enabled clients, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

An encrypted SSL connection requires all information sent between a client and a server to be encrypted by the sending software and decrypted by the receiving software, thus providing a high degree of confidentiality. This is important for both parties to any private transaction. In addition, all data sent over an encrypted SSL

connection is protected with a mechanism for detecting tampering (i.e. for automatically determining whether the data has been altered in transit).

3 Implementation Support

As the developer, you are not limited to any particular technology to access The API. For the purposes of this Implementation Guide, Postman will be used in all the examples.

3.1 The API Access Workflow

Accessing The API is a multi-step process (refer to Figure 1). The API only utilizes the POST verb for all operations.

The first step is to request an authorization token from the authentication server that will be included in all subsequent requests. The credentials needed to obtain the Authorization Token will be provided by the practice. An example of this step is shown in section (3.1.1 Authentication Token).

The next step is to send a request message (with the Authorization Token) that contains the Patient Identifiers, a start date, and end date to the endpoint (related to the data being requested) on the application server. The application server will then validate the Authorization Token If valid, the application server will return the data to the client. Examples of this step are shown in sections (3.2.2).

The Authorization Token has a time to live (TTL) of 60 minutes. There is no harm in always starting the process from Step 1 for every endpoint request.

Appendix B: Status Codes and Messages lists all status codes and status messages that may be return by The API.

3.1.1 Authentication Token

Authorization Endpoint Address:
https://localhost

Access Token Endpoint Address:
https://localhost //PulseAuthenticationToken/token/generate

Here is an example of requesting an Authorization Token from the authentication Server using Postman:

Pulse

Pulse Pro/Ehr (PulseUnifiedApiToken) / **Call Get Token (PulseAuthenticationToken)**

POST {{baseUrl}}/PulseAuthenticationToken/token/generate

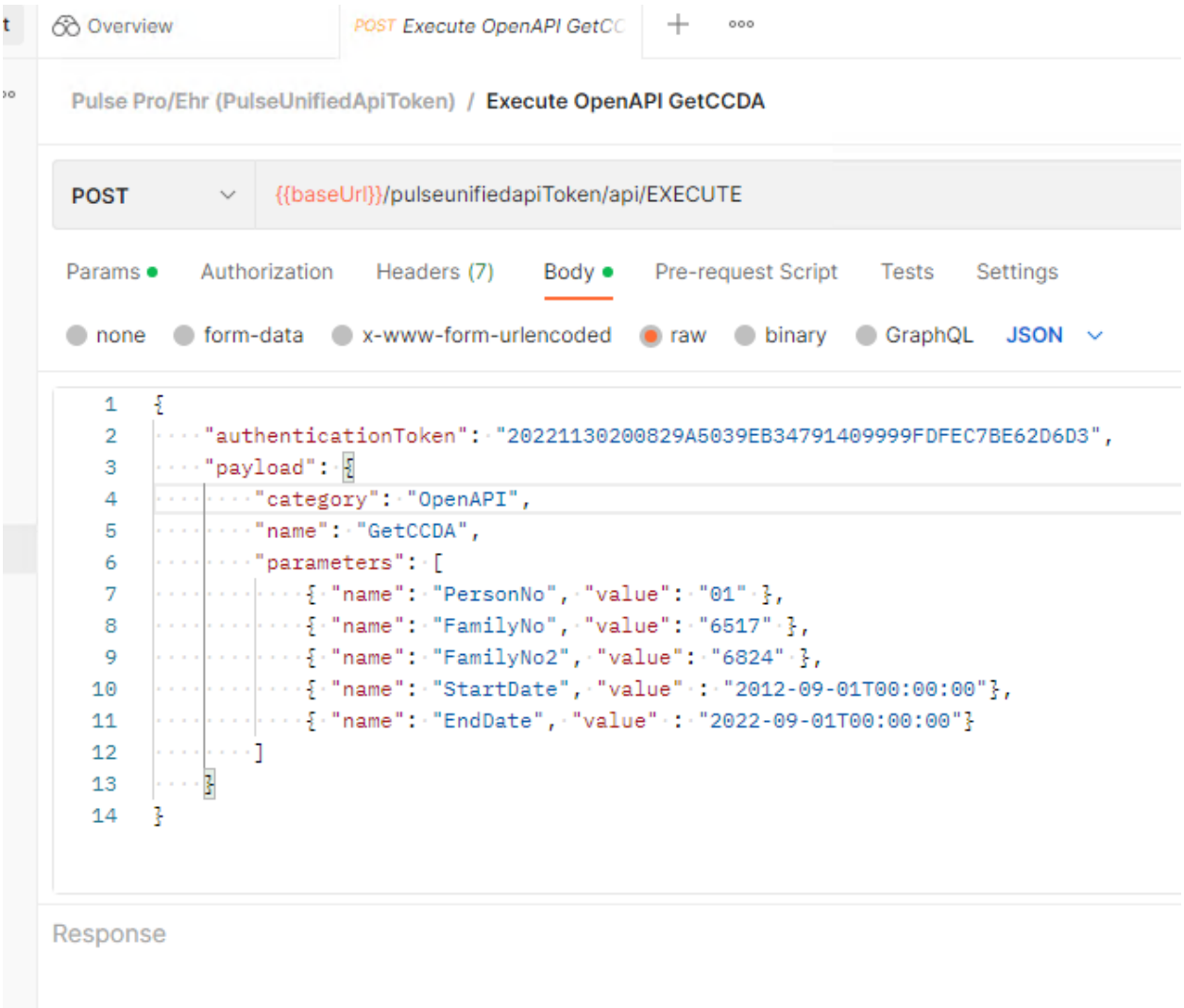
Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

☑	Authorization	Basic UFVMU0VfUEFUSUVOVF9QT1JUQ...
☑	Cache-Control	no-cache
☑	Postman-Token	<calculated when request is sent>
☑	Content-Length	0
☑	Host	<calculated when request is sent>
☑	User-Agent	PostmanRuntime/7.30.0
☑	Accept	*/*
☑	Accept-Encoding	gzip, deflate, br
☑	Connection	keep-alive
	Key	Value

Body | Cookies | Headers (7) | Test Results

The basic creds will be provided by the practice. Once the Authorization Token has been received from the server, it must be used in all subsequent requests. The token is to be included in the header of the messages. For example:

Pulse



3.2 The API Endpoint Example

3.2.1 CCD Download

Endpoint Address:
https://localhost/pulseunifiedapiToken/api/EXECUTE

Request Message:

```
{
    "authenticationToken": "20221130200829A5039EB34791409999FDFEC7BE62D6D3",
    "payload": {
        "category": "OpenAPI",
        "name": "GetCCDA",
        "parameters": [
            { "name": "PersonNo", "value": "XX" },
```

```
            { "name": "FamilyNo", "value": "XXXXX" },
            { "name": "StartDate", "value" : "2012-09-01T00:00:00"},
            { "name": "EndDate", "value" : "2022-09-01T00:00:00"}
        ]
    }
}
```

Response Message:

POST {{baseUrl}}/pulseunifiedapiToken/api/EXECUTE

Params • Authorization Headers (9) Body • Pre-request Script Tests Settings

Body Cookies Headers (9) Test Results Status: 200 OK Time: 2.01 s Size

Pretty Raw Preview Visualize JSON


```
{
    "AuthenticationToken": "",
    "MessageID": "",
    "RelatedMessageID": "",
    "MessageType": "PulseCommandExecuteResponse",
    "UtcTimestamp": null,
    "Payload": {
        "Success": true,
        "Response": {
            "VendorPersonId": "6517-01",
            "VendorProviderId": "",
            "VendorClinicalDocumentId": "",
            "VendorEncounterId": "",
            "VendorModifyUserId": "",
            "CdaFileName": "",
            "CdaFileDescription": "",
            "MimeType": "",
            "DocumentTemplate": 1,
            "SummaryType": 4,
            "IsReferral": false,
            "Bytes": null,
            "VendorId": {
                "EncryptedValue": ""
            },
            "VendorModifyDate": null
        },
        "Category": "OpenAPI",
        "Name": "GetCCDA",
        "Parameters": [
            {
                "Name": "PersonNo",
                "Value": "01"
            },
```


StartDate/EndDate Limitations:

If you provide a StartDate/EndDate, and the patient has no encounters in that period, then you will receive the following message back in place of a CCD:

“No encounter data was found; empty CCD will not be generated”

We must have an encounter to build a CCD from. The API will not return a blank CCD.

Note:
The contenst of the “bytes” property are base64 encoded. They will need to be decoded by the consuming application.

APPENDIX A: Common Clinical Data Components

The following table summarizes the endpoints for each component.

Component Name	Endpoint Address
Authorization Endpoint	https://localhost //PulseAuthenticationToken/token/generate
Get CCDA	https://localhost/pulseunifiedapiToken/api/EXECUTE

**Note: "localhost" is a placeholder for the actual domain name of the practice.*

APPENDIX B: Status Codes and Messages

The following table provides the Status Codes and Status Messages that may be returned to the client from The API.

Status Code	Message
200	Bytes property fully populated • This will contain a base64 encoded string for your XML CCD. • This is considered a successful response.
200	Authentication Token is Blank • This is an indication that your auth Token was not provided, or it has expired. Your application will need to re-authenticate, using the Get Token method.
200	Response = System.Argument Exception • This is an indication that you are not passing the 4 required arguments. PersonNo, FamilyNo, StartDate, EndDate • If you want an all encompassing CCD, please pass a date range that will include all the data you want. IE, 1900-1-1 to 2050-1-1
200	Decoded Bytes value is “No encounter data was found; empty CCD will not be generated” • This is an indication that you have provided a date range for the patient, where no encounter was found. Encounter data is required to build the CCD. Please alter your date range.
200	Response values are blank, and/or null Bytes value • This is an indication that the FamilyNo/PersonNo is not accurate.
500	Internal Server Error