HPMS
Health Plan Management System

Part D Pricing File Submission (PDPFS) Application Programming Interface (API) Technical Guide

Contents

High Level API Workflow

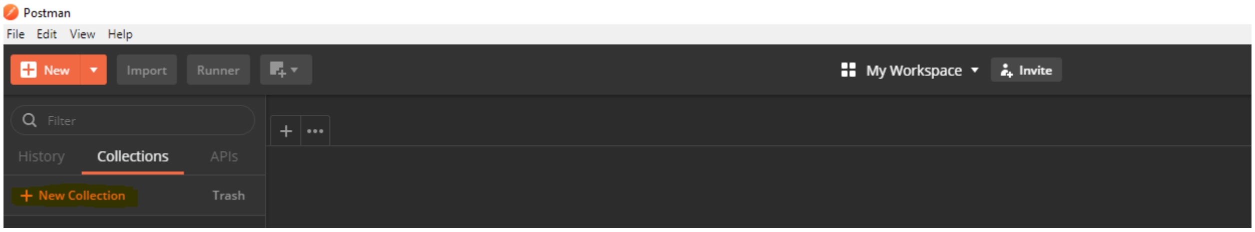
Step 1: Using the API key properties provided, the API client will send a request to the IDM service API to get an Authorization token (JWT). A CMS user ID is required in order to obtain the token. The token will be active for 60 minutes, and the process of requesting a new token can be automated by the client based on the expiration time.

Step 2: Using the Authorization token, the API client will make a POST call to the HPMS Upload endpoint. The Upload endpoint will verify the token and upload request, and if all business rules pass, the API will return an upload success with a confirmation ID or a failure response.

Step 3: Using the confirmation ID passed back from the Upload endpoint, the API client can make a request to obtain the submission status of that particular upload via the submission status endpoint.

Note: For questions regarding this document or technical issues with the API, users should contact hpmstechsupport@softrams.com.

Detailed Steps to Access the PDPFS File Upload and Submission Status Endpoints



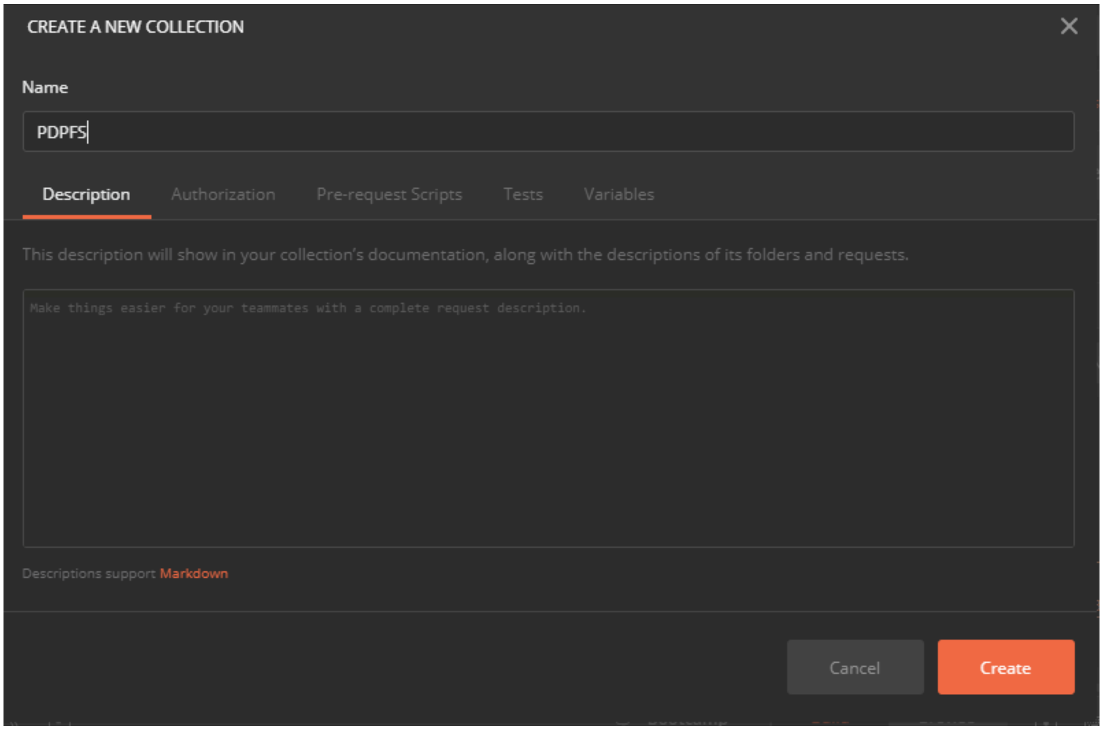
The following are **examples** of software options that can be used for calling the endpoint:

- Postman
- Fiddler
- SoapUI

Likewise, there are various programming languages that may be used. This document provides code snippets for NodeJS, cURL, and GoLang. **You may use whichever technical solutions work for your organization.**

Step 1: Retrieving an authorization token from the IDM service, including the steps using Postman.

1. Create and label a new collection. This document will refer to the collection as PDPFS going forward.


CREATE A NEW COLLECTION

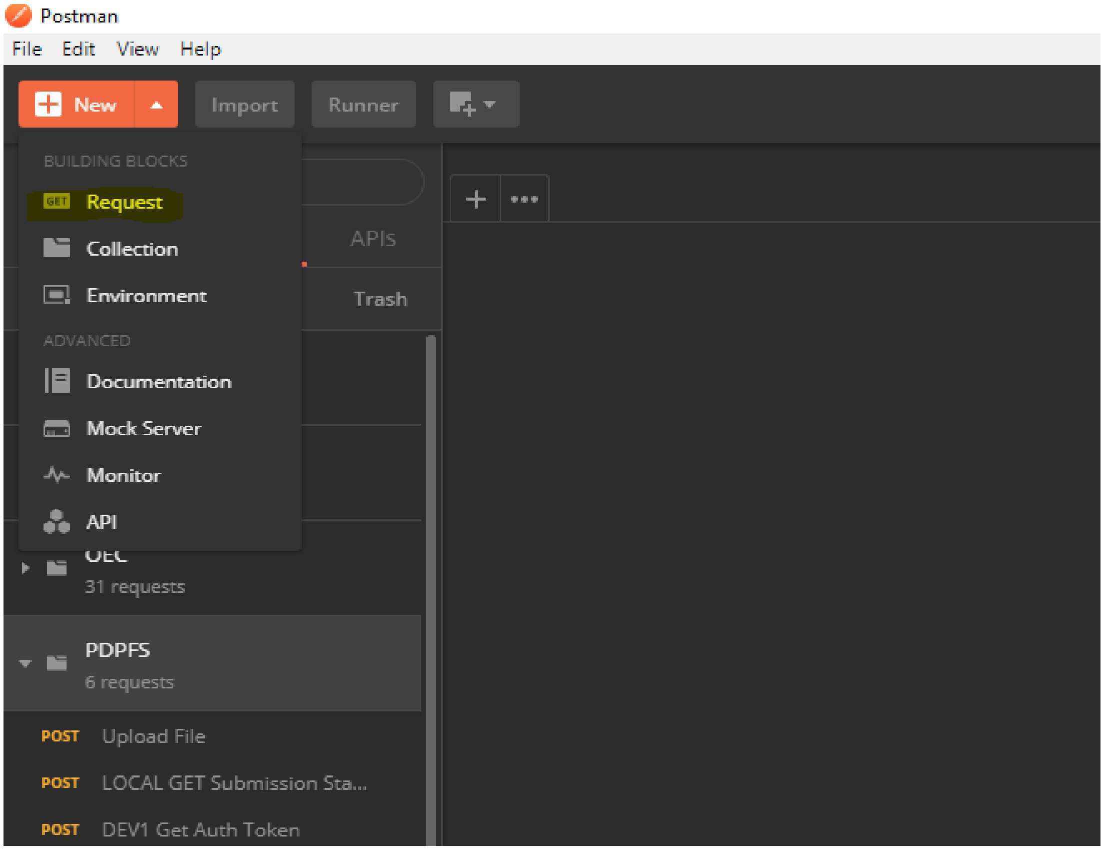
Name: PDPFS

This description will show in your collection's documentation, along with the descriptions of its folders and requests.

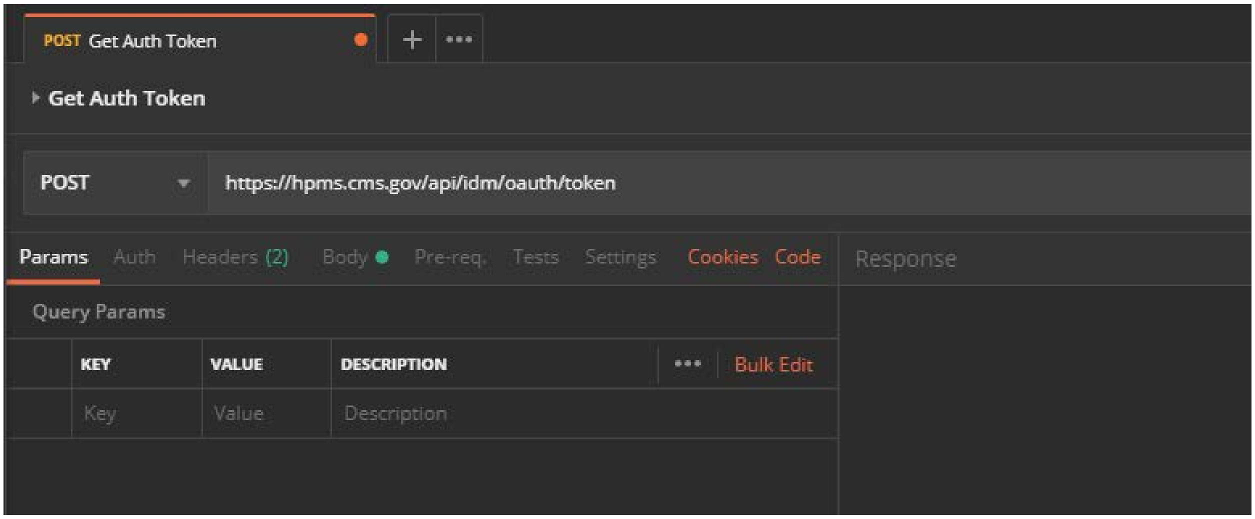
Make things easier for your teammates with a complete request description.

Descriptions support Markdown

2. Create a new request and save it to the PDPFS collection.

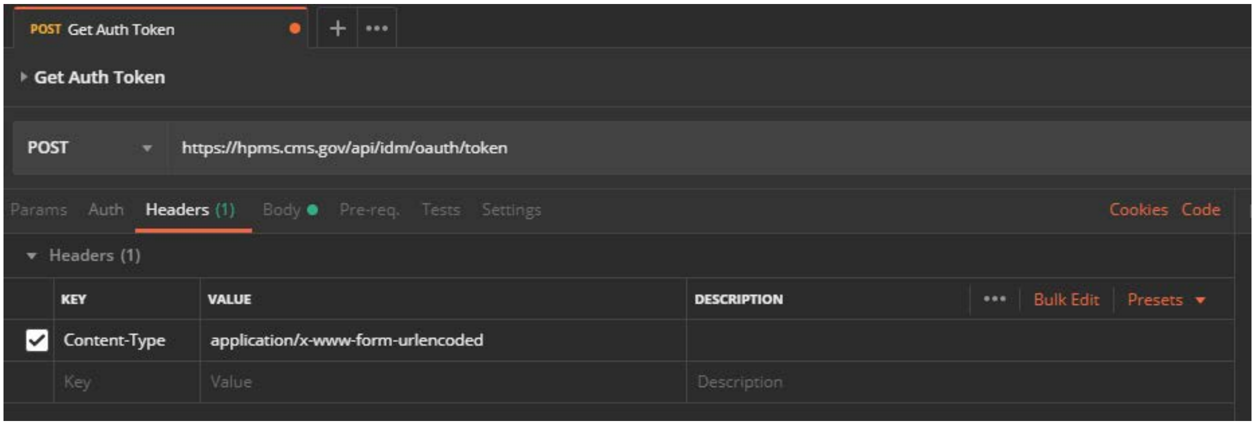


3. Change the type to POST and add the IDM endpoint URL.



POST https://hpms.cms.gov/api/idm/oauth/token

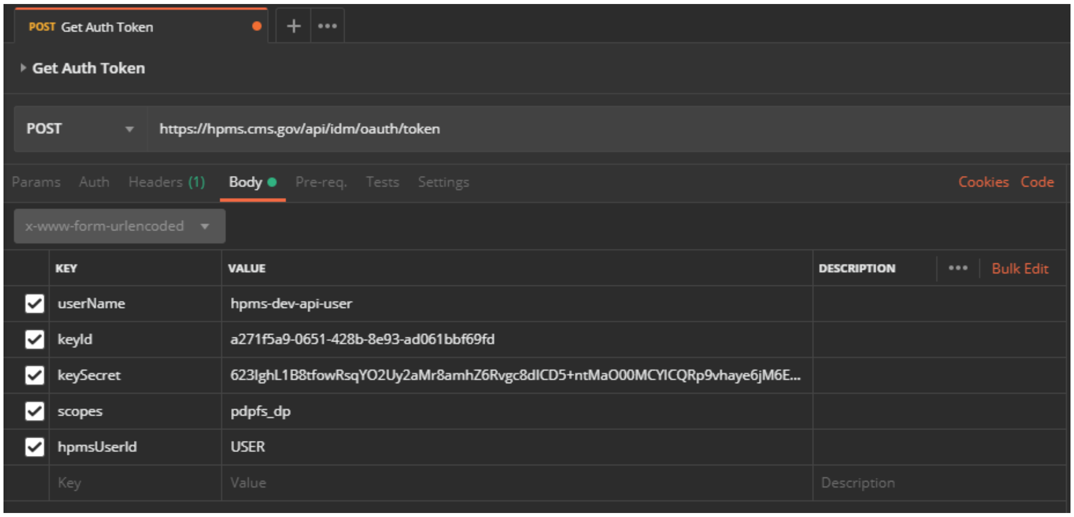
4. Add a header with a key of Content-Type and value of application/x-www-form-urlencoded.



POST https://hpms.cms.gov/api/idm/oauth/token

KEY	VALUE	DESCRIPTION
☑ Content-Type	application/x-www-form-urlencoded	

5. In the Body, add the following fields. The associated values will be supplied to you once you are granted access to use the endpoints.
    a. username
    b. keyId
    c. keySecret
    d. scopes
    e. hpmsUserId

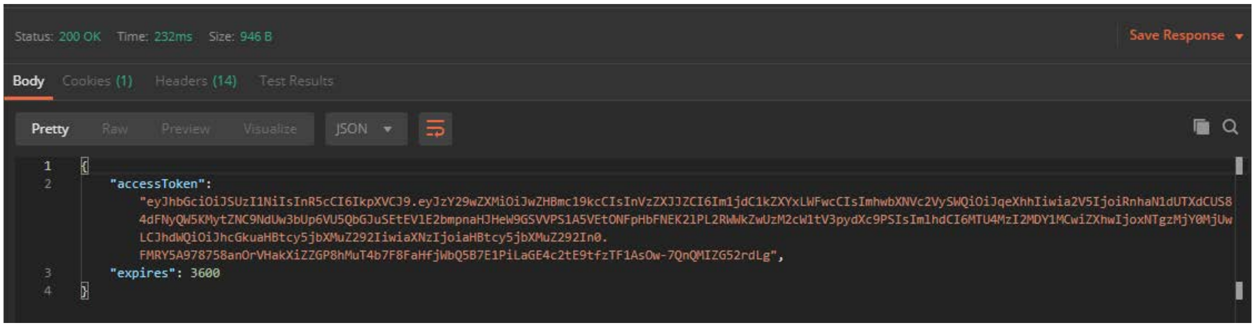


POST https://hpms.cms.gov/api/idm/oauth/token

x-www-form-urlencoded

	KEY	VALUE	DESCRIPTION
☑	userName	hpms-dev-api-user	
☑	keyId	a271f5a9-0651-428b-8e93-ad061bbf69fd	
☑	keySecret	623IghL1B8tfowRsqYO2Uy2aMr8amhZ6Rvgc8dICD5+ntMaO00MCYICQRp9vhaye6jM6E...	
☑	scopes	pdpfs_dp	
☑	hpmsUserId	USER	
	Key	Value	Description

6. Click the "Send" button. The response should provide you with a temporary access token which you will use in subsequent calls to the PDPFS API.



Status: 200 OK Time: 232ms Size: 946 B

```
{
    "accessToken":
        "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzY29wZXMiOiJwZHBmc19kcCIsInVzZXJJZCI6Im1jdC1kZXYxLWFwcCIsImhwbXNVc2VySWQiOiJqeXhhIiwia2V5IjoiRnhaN1dUTXdCUS8
        4dFNyQW5KMytZNC9NdUw3bUp6VU5QbGJuSEtEV1E2bmpnaHJHeW9GSVVPS1A5VEtONFpHbFNEK21PL2RWWkZwUzM2cW1tV3pydXc9PSIsImlhdCI6MTU4MzI2MDY1MCwiZXhwIjoxNTgzMjY0MjUw
        LCJhdWQiOiJhcGkuaHBtcy5jbXMuZ292IiwiaXNzIjoiaHBtcy5jbXMuZ292In0.
        FMRY5A978758anOrVHakXiZZGP8hMuT4b7F8FaHfjWbQ5B7E1PiLaGE4c2tE9tfzTF1AsOw-7QnQMIZG52rdLg",
    "expires": 3600
}
```

IDM Code Snippets Examples

NodeJs (using Request):

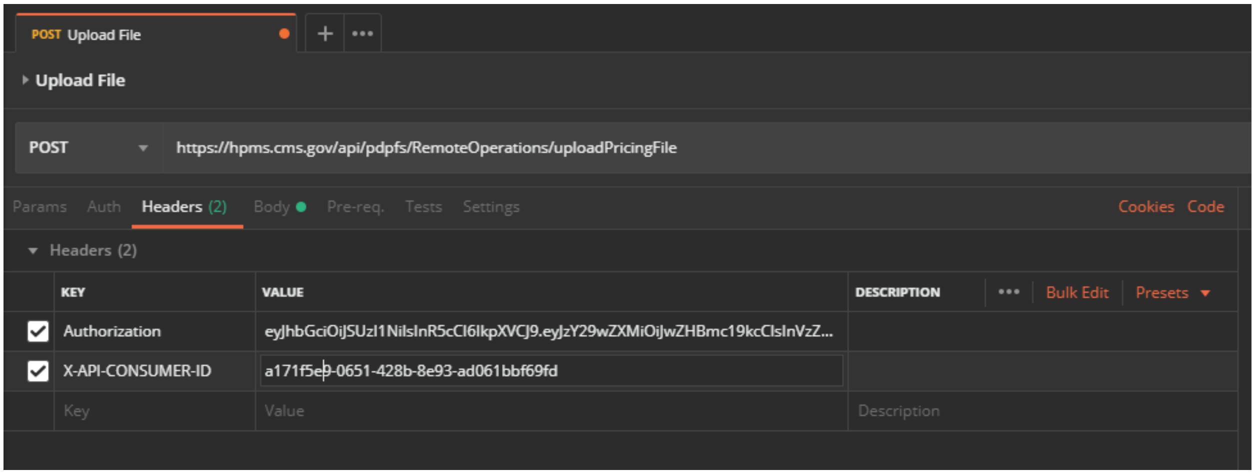
```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'https://hpms.cms.gov/api/idm/oauth/token',
  'headers': {
    'Content-Type': 'application/x-www-form-urlencoded'
  },
  form: {
    'userName': 'mct-dev-api-user',
    'keyId': 'a171f5e9-0651-428b-8e93-ad061bbf69fd',
    'keySecret':
'623IghL1B8teowRsqYO2Uy2aMr3amhZ6Rvgc8dICD5+ntMaO00MCYICQRp9vhaye6jM6EyREiJdpvcXmCC13Qw==',
    'scopes': 'pdpfs_dp',
    'hpmsUserId': 'USER'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

cURL:

```
curl --location --request POST 'https://hpms.cms.gov/api/idm/oauth/token' \

--header 'Content-Type: application/x-www-form-urlencoded' \

--data-urlencode 'userName=mct-dev-api-user' \

--data-urlencode 'keyId=a171f5e9-0651-428b-8e93-ad061bbf69fd' \

--data-urlencode
'keySecret=623IghL1B8teoeRsqYO2Uy2aMr8amhZ6Rvgc8dICD5+ntMaO00MCYICQRp9vhaye6jM6EyREiJdpvcXmCC13Qw
==' \

--data-urlencode 'scopes=pdpfs_dp' \

--data-urlencode 'hpmsUserId=USER
```

Step 2: Uploading the pricing files (PC, PF, and optional CP file)

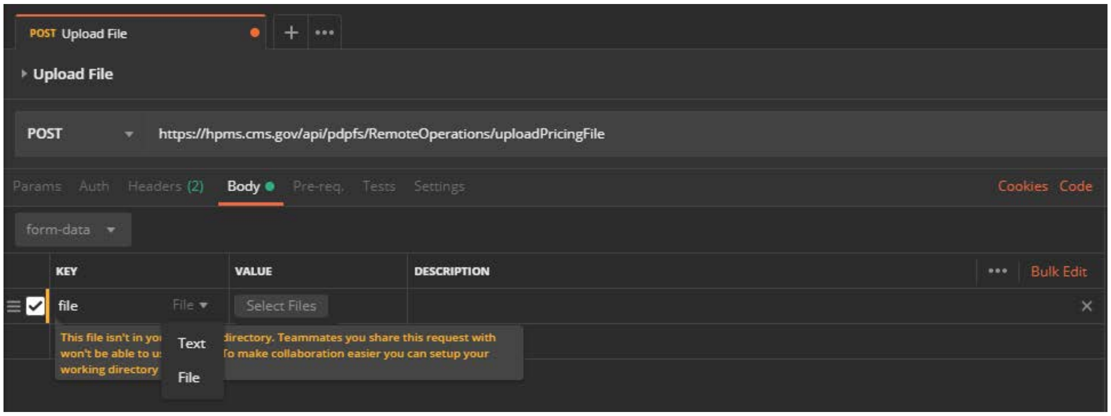
1. Create a new request POST request and add the upload pricing file endpoint.
2. Add the following headers:
    a. Authorization: This will be the value of the authorization token you retrieve from the previous IDM call.
    b. X-API-CONSUMER-ID: This is the keyId from the previous IDM call which will be supplied to you when you are granted access to the endpoints.
    c. Contractyear: The contract year you wish to submit for.
    d. ContractId: The ID of the contract you are submitting for.



POST https://hpms.cms.gov/api/pdpfs/RemoteOperations/uploadPricingFile

	KEY	VALUE	DESCRIPTION
☑	Authorization	eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzY29wZXMiOiJwZHBmc19kcCIsInVzZ...	
☑	X-API-CONSUMER-ID	a171f5e9-0651-428b-8e93-ad061bbf69fd	
	Key	Value	Description

3. In the body add a key called file and select file from the drop-down box that appears to the right of the text box.



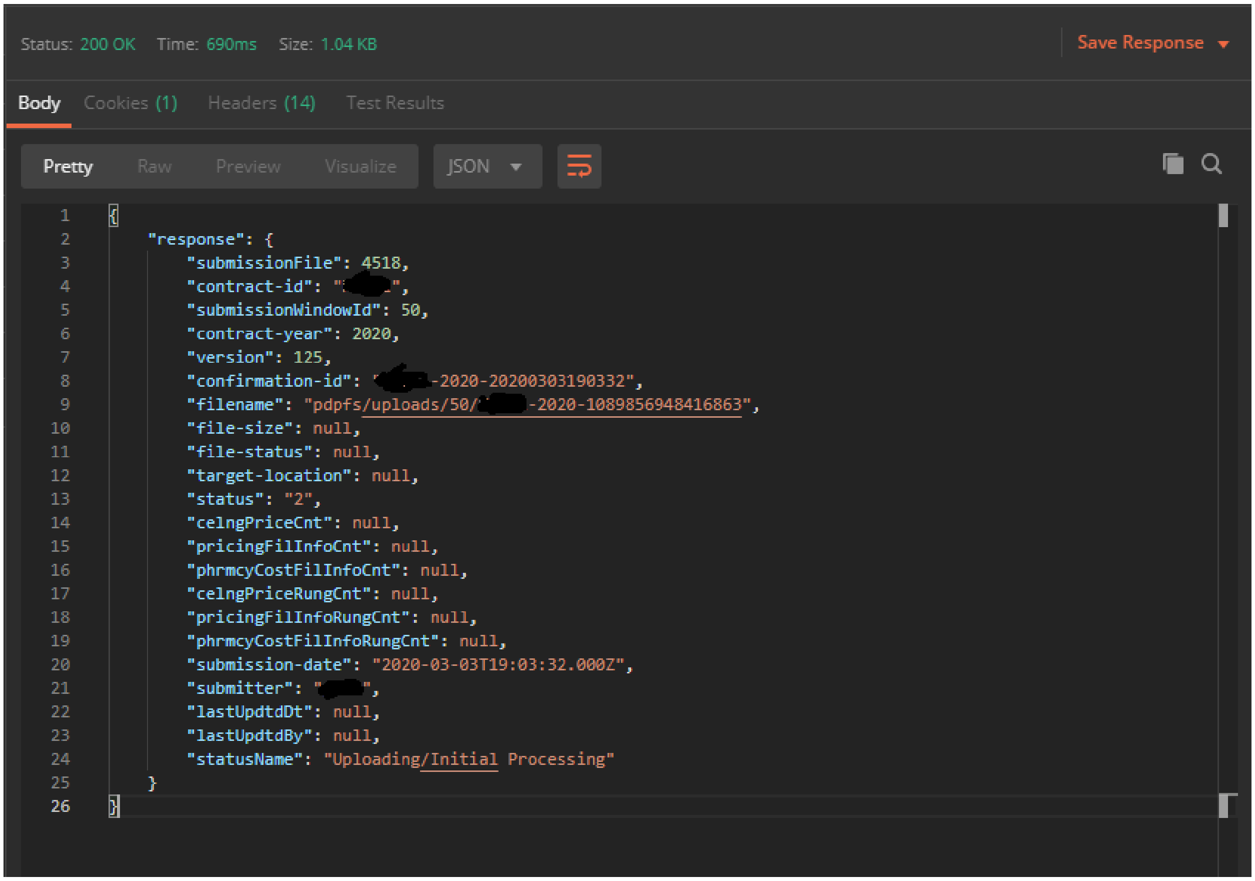
POST https://hpms.cms.gov/api/pdpfs/RemoteOperations/uploadPricingFile

form-data

KEY	VALUE	DESCRIPTION
☑ file (File)	Select Files	

This file isn't in yo directory. Teammates you share this request with
won't be able to u: To make collaboration easier you can setup your
working directory

4. Press the “Select Files” button in the value column and navigate to the pricing file you wish to upload and select it. Press send and you will receive a similar response as shown below.



Status: 200 OK Time: 690ms Size: 1.04 KB

```
{
    "response": {
        "submissionFile": 4518,
        "contract-id": "[redacted]",
        "submissionWindowId": 50,
        "contract-year": 2020,
        "version": 125,
        "confirmation-id": "[redacted]-2020-20200303190332",
        "filename": "pdpfs/uploads/50/[redacted]-2020-1089856948416863",
        "file-size": null,
        "file-status": null,
        "target-location": null,
        "status": "2",
        "celngPriceCnt": null,
        "pricingFilInfoCnt": null,
        "phrmcyCostFilInfoCnt": null,
        "celngPriceRungCnt": null,
        "pricingFilInfoRungCnt": null,
        "phrmcyCostFilInfoRungCnt": null,
        "submission-date": "2020-03-03T19:03:32.000Z",
        "submitter": "[redacted]",
        "lastUpdtdDt": null,
        "lastUpdtdBy": null,
        "statusName": "Uploading/Initial Processing"
    }
}
```

Code Snippet Examples for Uploading the Pricing Files

NodeJS

```
var request = require('request');
var fs = require('fs');
var options = {
  'method': 'POST',
  'url': 'https://hpms.cms.gov/api/pdpfs/RemoteOperations/uploadPricingFile',
  'headers': {
    'Authorization':
'eyJhbGcieiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzY29wZXMiOiJwZHBmc19kcCIsInVzZXJJZCI6Im1jdC1kZXYxLWFwcCIsImhwbXN
Vc2VySWQiOiJqeXhhIiwia2V5IjoiRnhaN1dUTXdCUS84dFNyQW5KMytZNC9NdUw3bUp6VU5QbGJuSEtEVlE2bmpnaHJHe
W9GSVVPS1A5VEtONFpHbFNEK2lPL2RWWkZwUzM2cW1tV3pydXc9PSIsImlhdCI6MTU4MzI2MjAwMSwiZXhwIjoxNTgzMj
Y1NjAxLCJhdWQiOiJhcGkuaHBtcy5jbXMuZ292IiwiaXNzIjoiaHBtcy5jbXMuZ292In0.qGBzhAg9AUU8yKSR7E6n-
6fHToy4NlLGqfRgSAILTT36TS7s40Qh0whg0qE8Y8e1lLXRKO66vwF2YhtNPPbblA',
    'X-API-CONSUMER-ID': 'a171f5e9-0651-428b-8e93-ad061bbf69fd',
    'contractyear': '2020',
    'contractid': 'H0001',
    'Content-Type': 'multipart/form-data; boundary=--------------------------428460616493473796797408'
  },
  formData: {
    'file': {
      'value': fs.createReadStream('/C:/PATH/TO/FILE/H0001.zip'),
      'options': {
        'filename': '/C:/PATH/TO/FILE/H0111.zip',
        'contentType': null
      }
    }
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

cURL

```
curl --location --request POST 'https://hpms.cms.gov/api/pdpfs/RemoteOperations/uploadPricingFile' \

--header 'Authorization:
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXeCJ9.eyJzY29wZXMiOiJwZHBmc19kcCIsInVzZXJJZCI6Im1jdC1kZXYxLWFwcCIsImhwbXN
Vc2VySWQiOiJqeXhhIiwia2V5IjoiRnhaN1dUTXdCUS84dFNyQW5KMytZNC9NdUw3bUp6VU5QbGJuSEtEVlE2bmpnaHJHe
W9GSVVPS1A5VEtONFpHbFNEK2lPL2RWWkZwUzM2cW1tV3pydXc9PSIsImlhdCI6MTU4MzI2MjAwMSwiZXhwIjoxNTgzMj
Y1NjAxLCJhdWQiOiJhcGkuaHBtcy5jbXMuZ292IiwiaXNzIjoiaHBtcy5jbXMuZ292In0.qGBzhAg9AUU8yKSR7E6n-
6fHToy4NlLGqfRgSAILTT36TS7s40Qh0whg0qE8Y8e1lLXRKO66vwF2YhtNPPbblA' \

--header 'X-API-CONSUMER-ID: a171f5e9-0651-428b-8e93-ad061bbf69fd' \

--header 'contractyear: 2020' \

--header 'contractid: H0001' \

--header 'Content-Type: multipart/form-data; boundary=--------------------------428460616493473796797408' \

--form 'file=@/C:/PATH/TO/FILE/H0001.zip'
```

Step 3: Checking the submission status of a file

1. Create a new request POST request and add the check submission status endpoint (https://hpms.cms.gov/api/pdpfs/RemoteOperations/checkSubmissionStatus)
2. Add the following headers:
   a. Authorization: This will be the value of the authorization token you retrieve from the previous IDM call.
   b. X-API-CONSUMER-ID: This is the keyId from the previous IDM call which will be supplied to you when you are granted access to the endpoints.
3. In the body select raw and then change the drop down from Text to JSON
4. Add the list of confirmation numbers to the body using the following pattern
   a. Note you may submit more than one confirmation number at a time

```
{
   "confirmationNumbers": [
            " CONFIRMATION_NUMBER_1",
             “CONFIRMATION_NUMBER_2”,
            “CONFIRMATION_NUMBER_3”, …
            “CONFIRMATION_NUMBER_X”
   ]
}
```

Error Handling / Submission Status Messages

The following error messages are currently delivered via the HPMS PDPFS module. Additional error messages may be added based on user experience during the pilot. Each error message with its corresponding input fields are mapped for easy reference. This is the current list, but it is subject to change.

Required Fields	Response Messages
Headers	
Authorization	When missing or invalid:

```
{
   "error": {
      "statusCode": 401,
      "name": "Error",
      "message": "Authorization Required",
      "code": "AUTHORIZATION_REQUIRED"
   }
}
```

X-API-CONSUMER-ID	When missing:

```
{
   "error": {
      "statusCode": 401,
      "name": "Error",
      "message": "Authorization Required",
      "code": "AUTHORIZATION_REQUIRED"
   }
}
```

contractYear	When missing:

```

{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "Missing contract year"
   }
}
```

contractId	When missing:

```

{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "Missing contract id"
   }
}
```

Not authorized to operate on contract:

```
{
   "error": {
      "statusCode": 401,
      "name": "Unauthorized",
      "message": "API user: cms-dev1-app associated with HPMS user: jyxa is not authorized to operate on contract: R1234. "
   }
}
```

Body	
file	When missing:

```
{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "No file detected"
   }
}
```

ContractId doesn't match file name:

```
{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "ContractId: R1234 does not match name of file: H4321"
   }
}
```

File is not a Zip

```
{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "Incorrect file type: text/plain detected application/zip is required"
   }
}
```

File name is incorrectly formatted

```
{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "File name: fileName not in proper format"
   }
}
```

File too large

```
{
   "error": {
      "statusCode": 400,
      "name": "Bad Request",
      "message": "File too large"
   }
}
```

General Msgs

Submission window

Good request

```
{
   "response": {
      "submissionFile": 4501,
      "contract-id": "R1234",
      "submissionWindowId": 50,
      "contract-year": 2020,
      "version": 108,
      "confirmation-id": "R1234-2020-20200212160548",
      "filename": "pdpfs/uploads/50/R1234-2020-9176610945964541",
      "file-size": null,
      "file-status": null,
      "target-location": null,
      "status": "2",
      "celngPriceCnt": null,
      "pricingFillInfoCnt": null,
      "phrmcyCostFillInfoCnt": null,
      "celngPriceRungCnt": null,
      "pricingFillInfoRungCnt": null,
      "phrmcyCostFillInfoRungCnt": null,
      "submission-date": "2020-02-12T16:05:48.000Z",
      "submitter": "jyxa",
      "lastUpdtdDt": null,
      "lastUpdtdBy": null,
      "statusName": "Uploading/Initial Processing"
   }
}
```

	Contract did not submit in window 1

```
{
   "error": {
      "statusCode": 401,
      "name": "Unauthorized",
      "message": "Contract did not submit in window 1."
   }
}
```

	Window 1 submission not in proper status

```
{
   "error": {
      "statusCode": 401,
      "name": "Unauthorized",
      "message": "Window 1 submission is not in review required or resubmission required status."
   }
}
```

Status Check

Required Fields	Response Messages
Headers	
Authorization	When missing or invalid:

```
{
   "error": {
      "statusCode": 401,
      "name": "Error",
      "message": "Authorization Required",
      "code": "AUTHORIZATION_REQUIRED"
   }
}
```

X-API-CONSUMER-ID	When missing or invalid:

```
{
   "error": {
      "statusCode": 401,
        "name": "Error",
        "message": "Authorization Required",
        "code": "AUTHORIZATION_REQUIRED"
    }
}
```

Body

Confirmation Numbers

Good request:

```
{
    "submissionStatuses": [
        {
            "ConfirmationNumber": "R1234-2020-20200129154043",
            "SubmissionStatus": "Level Two Validation - Fatal Errors"
        }
    ]
}
```

Good request:

```
{
    "submissionStatuses": [
        {
            "ConfirmationNumber": "R1234-2020-20200129154043",
            "SubmissionStatus": "Level Two Validation - Fatal Errors"
        },
        {
            "ConfirmationNumber": "R1234-2020-20200212141659",
            "SubmissionStatus": "Uploading/Initial Processing"
        }
    ]
}
```

Not authorized to access requested contract:

```
{
    "error": {
        "statusCode": 401,
        "name": "Unauthorized",
        "message": "User not authorized to retrieve status for contractIds: R1234"
    }
}
```

Confirmation number not found:

```
{
   "submissionStatuses": [
      {
         "ConfirmationNumber": "H0111-2020-2020012914043",
         "SubmissionStatus": "Confirmation number does not exist"
      }
```