

G. BDDS REST API Documentation

This document describe the way to use the Rest API for the BDDS application. Rest API exposed here will give the repsonse in the Json .

On the API, if the list of product endpoints are called without the token, it will only return the list of public product and if the valid token is passed, it will return the aggregated list of public and subscribed products for the corresponding user

Below is the set up instructions which will help with all the details for the API

1. Authentication

Prior to any action to get the list of subscribed products and to download, it is required to authenticate via the UserId/Password credentials to get the valid authentication token which is valid for **60 mins**. Authentication is based on a standard OAuth 2 grant type: Password Credentials. End User will need to use his credential's and need to make a call to the OKTA URL to get the proper token. Bearer token that is to be returned by the OKTA authentication endpoint will be valid for 1 hours. After expiry time of such token , end user need to request a new token.

Type : oauth2

Flow : application

Token URL : <https://login.epo.org/oauth2/aus3up3nz0N133c0V417/v1/token>

Here user name would be the EPO Email ID (Example as akhtar1508@gmail.com) and password would be your login password

To get the OKTA token , it requires to have the valid client ID (**0oa3updn7an5pMI8O417**) in the of base 64 encoded format . Base 64 encoded value of given client ID is **MG9hM3VwZG43YW41cE1JOE80MTc=** and same has to be passed in Authorization header along as Basic authentication as mentioned as part of CURL command .

Corresponding CURL Command :

```
curl --location --request POST 'https://login.epo.org/oauth2/aus3up3nz0N133c0V417/v1/token' \
--header 'Authorization: Basic MG9hM3VwZG43YW41cE1JOE80MTc=' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'username=*****' \
--data-urlencode 'password=*****' \
--data-urlencode 'scope=openid'
```

Another way of Doing in Postman

(In the Authorization header need to pass the value as **Basic MG9hM3VwZG43YW41cE1JOE80MTc=**)

POST

https://login.epo.org/oauth2/aus3up3nz0N133c0V417/v1/token ...

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION		Bulk Edit
<input checked="" type="checkbox"/>	grant_type	password			
<input checked="" type="checkbox"/>	username	akhtar1508@gmail.com			
<input checked="" type="checkbox"/>	password				
<input checked="" type="checkbox"/>	scope	openid			
	Key	Value	Description		

Body

Cookies (1)

Headers (23)

Test Results

200 OK 1099 ms 2.65 KB

Save Response

code

Sample JAVA Program to Get the authentication Token

code

```

public final static String USERNAME = "*****"; // need to pass user own EPO email ID
public final static String PASSWORD = "*****"; // need to pass user own EPO password
public final static String TOKEN_URL = "https://login.epo.org/oauth2/aus3up3nz0N133c0V417/v1/token";
public final static String CLIENT_ID = "0oa3updn7an5pMI8O417";

private static TokenResponse getAccessToken() {
    MultiValueMap<String, String> body = new LinkedMultiValueMap<> ();
    body.put("grant_type", Collections.singletonList("password"));
    body.put("username", Collections.singletonList( USERNAME )); // OKTA Email ID
    body.put("password", Collections.singletonList( PASSWORD )); // OKTA Password
    body.put("scope", Collections.singletonList("openid"));
    ResponseEntity<TokenResponse> responseEntity =
        new RestTemplate ().exchange(
            TOKEN_URL, // Token URL
            HttpMethod.POST,
            new HttpEntity<Object> (body, getTokenForPasswordFlow()),
            TokenResponse.class);
    TokenResponse res = responseEntity.getBody();
    return res;
}

private static HttpHeaders getTokenForPasswordFlow() {
    HttpHeaders httpHeaders = new HttpHeaders();
    httpHeaders.setContentType( MediaType.APPLICATION_FORM_URLENCODED);
    byte[] encodedAuth = Base64.getEncoder ().encode ( CLIENT_ID.getBytes () ); // Application Client ID
    String authHeader = "Basic " + new String(encodedAuth, Charset.defaultCharset());
    httpHeaders.set("Authorization", authHeader);
    return httpHeaders;
}

```

Once the token is granted, these token must be used in the subsequent request as **Bearer token** and should be passed in as Authorization header

2. To get the list of products

GET <https://publication-bdds.apps.epo.org/bdds/bdds-bff-service/prod/api/products/>

Note: Authorization header must be passed with valid token in the request header, so that list of subscribed product can be assessed. If the token is not passed, above endpoint will just return the public products

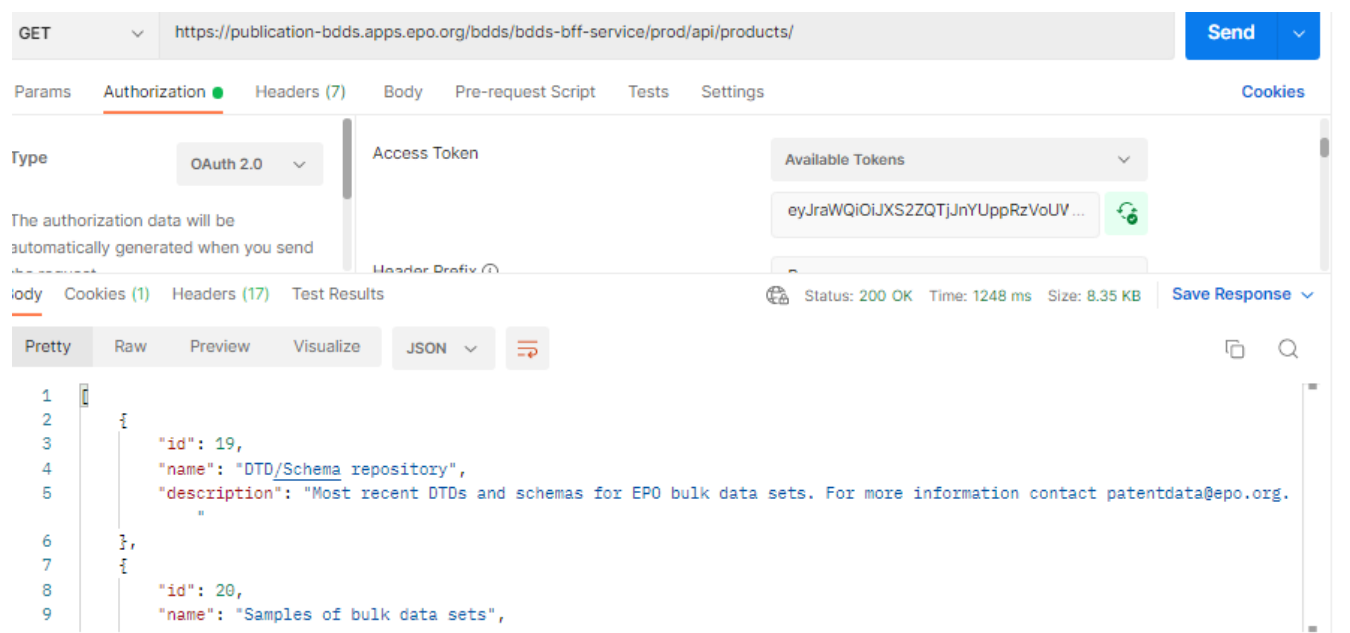
Ex: Authorization = **Bearer** eyJraWQiOiJXS2ZQTjJnYUppRzVoUWlWZDnrc3oyY2E2dFB6T1FyZ2xLMVVLN2ITNDQ0liwiVWxnIjoiUIMyNTYiIq.
eyJ2ZXI0IjEslmp0aSI6IklFULkpJQUUp2aG1Sb2JrIjVlY292dFdnN1hNeWpCv0gtN1d2U0lzSm1mNnRhbFkiLCJpc3MiOiJodHRwczN2L2xvZ2UuLmVwby5vc
mc2b2F1dGgyL2F1czN1dCnuejBOMTMyZyBWNlE6J3liwiYXVkljo1YXB0e18vYmRkcylzImhldClm6TMTY1NDY5NjQ2NywiZSwhbwjXU0N2U0NzAwMDY3LCJjaWQ
iOiIwbnZlZ2ZkXkbjbjhwIjVtUk41Z3N1YnlsbnVpZCJl6awTdN3Y3YXN1ZG40Zm9uNDI3liwiZmVpbnRwZmV5Z2U0LjCjJhdCJhdXRoX3R3bnVUoiEjE2NTQ2OTY
0NjcslnN1Yil6ImFraHRhcjE1MDhAZ21haWwvY29tln0.GCCbwYZNuNqibg4kiAA6HaGxzOcXQRNEU-RYZkLJ8IYT09xDPfXK-vLSJwlqV96kMfkoQuCuD-
C4aM4xr6Til55gxDr1q1D1DBfwgja7OKID09RAq85C9Miq9LJKqmvVgWuNbSn1T6ln91x3A411MN1JPbpDdVngch9p49uM4izUJnWPBo7bkQuqcsVw35-
J9-a1u6lJS74Hf_UTWubLxe3ngyKOilFW8hNkyQfGSN43qHw-
hkW7fvZ2s376VOMfnyvliM_oilYhORcNylKw8KScJwcVqJnE0X8mtoJ6zfA3UraN5Klhxq1Ua4rmVQDrl6EWM17eKJefp3-A

Response from this would be like below

code

```
[
  {
    "id": 19,
    "name": "DTD/Schema repository",
    "description": "Most recent DTDs and schemas for EPO bulk data sets. For more information contact patentdata@epo.org."
  },
  {
    "id": 20,
    "name": "Samples of bulk data sets",
    "description": "This section contains EPO bulk data product samples. For more information contact patentdata@epo.org."
  },
  {
    "id": 12,
    "name": "Statistical mapping: CPC to FI, FI to CPC and CPC to IPC",
    "description": "The tables are produced by using a statistical analysis of symbols allocated to families of documents present in DOCDB and classified simultaneously in the CPC, IPC and FI schemes. \nSame content is also available also here: https://www.epo.org/searching-for-patents/helpful-resources/first-time-here/classification/cpc/cpc-fi.html\nhttps://www.epo.org/searching-for-patents/helpful-resources/first-time-here/classification/cpc/fi-cpc.html\nepo.org/searching-for-patents/helpful-resources/first-time-here/classification/cpc/ipccpc.html\n\nFor more information contact patentdata@epo.org. \n"
  },
  {
    "id": 28,
    "name": "Unitary Patent (UP) sample packages",
    "description": "See README file in each package for more details.\nWe will be adding packages from all bulk data products as we receive them. \nFor any questions contact support@epo.org\n\n"
  }
]
```

Similar Example in Postman to get all the list of subscribed products



Similar Example with Curl Command

3. To get the details of product along with delivery and files

<https://publication-bdds.apps.epo.org/bdds/bdds-bff-service/prod/api/products/19>

code

```
{
  "id": 19,
  "name": "DTD/Schema repository",
  "description": "Most recent DTDs and schemas for EPO bulk data sets. For more information contact
patentdata@epo.org.",
  "deliveries": [
    {
      "deliveryId": 106,
      "deliveryName": "DOCDB in XML - DTDs - exchange-documents-v2.5.8 ",
      "deliveryPublicationDatetime": "2022-06-07T19:11:00.000+02:00", // // UTC format
      "deliveryExpiryDatetime": null,
      "files": [
        {
          "fileId": 199,
          "fileName": "exchange-documents-v2.5.8.xsd",
          "fileSize": "544.6 kB",
          "fileChecksum": "0387391B23B683309C58EDDA791F4483D207ABDC",
          "filePublicationDatetime": "2022-06-07T19:11:00.000+02:00" // UTC format
        }
      ]
    }
  ]
}
```

4. To Download the item for corresponding delivery and product

GET <https://publication-bdds.apps.epo.org/bdds/bdds-bff-service/prod/api/products/{productID}/delivery/{deliveryID}/file/{fileID}/download>

<https://publication-bdds.apps.epo.org/bdds/bdds-bff-service/prod/api/products/19/delivery/106/file/199/download>

It will generate the Response and will be sent as streaming and it needs to be save on the client end.

Note:

If you pass the invalid Bearer token or expired token, all the API's will return the HTTP response code as **401** and in this scenario, you need to generate the new token programmatically and pass it in the API.

If you call the API's without these token, API's will return the Response only for the public product.

If you try to get the product details for the subscribed products without the proper Bearer token, API will return the 401 response code with message as "Access Denied"