CONNECTING VIA API

Product Enrichment Portal

MY STORE

GETTING STARTED

TEST ENVIRONMENT

- Send request to Myer team on your decision to enrich via API
- The Online Enrichment Team will create a User profile in the PEP Test environment
- Suppliers will be sent API credentials via email with Key and Secret Code
- Suppliers can then connect and trial enriching products via API in the test environment
- Once successfully tested, Myer team will provide API credentials for the production environment

PEP ENRICHMENT STATUS

Setting the correct 'enrichment status' of the Product will ensure you're sending the right details to us via API.

When enriching products, follow the below important rules -

- API call for image status 10 and follow image enrichment to Myer guidelines
- API call for copy status 10 and follow copy enrichment (product names, descriptions, attributes) to Myer guidelines
- Update mandatory attributes – can be pulled via API but can be provided upon request Set status to 20 after enrichment

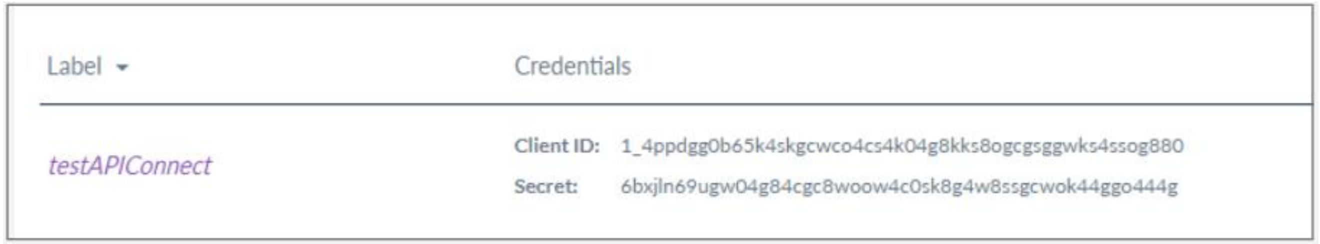
MYER MY STORE

IMPORTANT CONSIDERATIONS

- Suppliers should not send enrichment updates for products with status 20, as this will override any optimisation Myer has performed before it was published on the website
- Suppliers should not send enrichment updates for products marked 'Myer Image Status 30' or 'Copy Status 30' - these are products that have been rejected and need to be corrected before resending
- Please avoid sending a single large batch of products or making too many API calls at once. Limit is a max of 20 API calls per minute. In 'update API', please send a max of 50 products in one update call
- Suppliers enriching via API will be tiered as GOLD. Hence, images submitted are required to have a dimension of **1551x2000px**
- For bulk listing rejections, your welcome to send the corrected copy via API for the incorrect attribute. However, for lesser number of products, we recommend using the Bulk upload feature or correcting it directly in PEP

API TECHNICAL SPECS

1. GENERATING Base64



Label	Credentials
testAPIConnect	**Client ID:** 1_4ppdgg0b65k4skgcwco4cs4k04g8kks8ogcgsggwks4ssog880 **Secret:** 6bxjln69ugw04g84cgc8woow4c0sk8g4w8ssgcwok44ggo444g

Base64 token need to be generated place “:” between “Client ID” and “Secret Key” and UTF -8 Encoding

1_4ppdgg0b65k4skgcwco4cs4k04g8kks8ogcgsggwks4ssog880:6bxjln69ugw04g84cgc
8woow4c0sk8g4w8s sgcwok44ggo444g

Generated Base64 Token

MV80cHBkZ2cwYjY1azRza2djd2NvNGNzNGswNGc4a2tzOG9nY2dzZ2d3a3M0c3NvZzg4MDo
2YnhqbG42OXVndzA0Zzg0Y2djOHdvb3c0YzBzazhnNHc4c3NnY3dvazQ0Z2dvNDQ0Zw==

API TECHNICAL SPECS

2. USING Base64

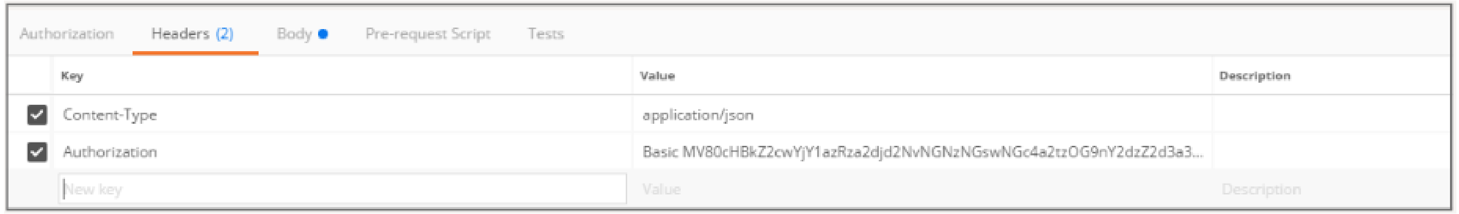
Generated Base64 should be used as “Authorization” Header while calling authentication method. Base64 Token need to be prefixed with “Basic”

For Example

Basic

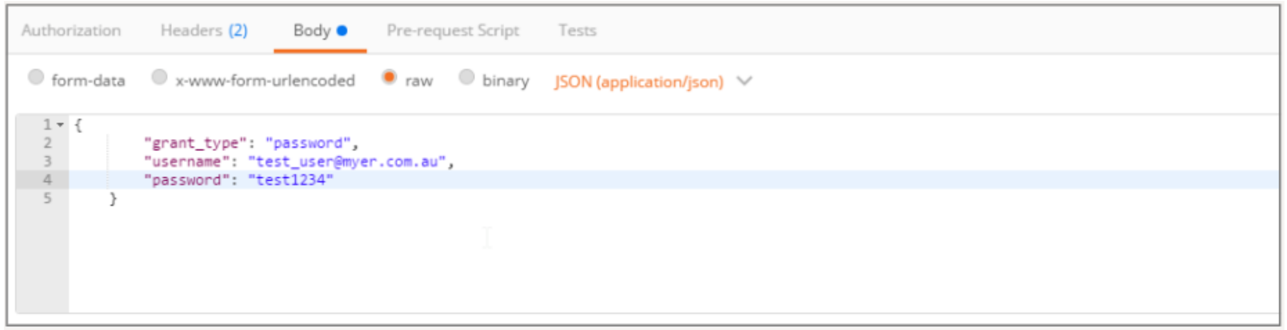
MV80cHBkZ2cwYjY1azRza2djd2NvNGNzNGswNGc4a2tzOG9nY2dzZ2d3a3M0c3NvZzg4MDo 2YnhqbG42OX VndzA0Zzg0Y2djOHdvb3c0YzBzazhnNHc4c3NnY3dvazQ0Z2dvNDQ0Zw==

Example (Request)



Authorization | Headers (2) | Body ● | Pre-request Script | Tests

	Key	Value	Description
✓	Content-Type	application/json	
✓	Authorization	Basic MV80cHBkZ2cwYjY1azRza2djd2NvNGNzNGswNGc4a2tzOG9nY2dzZ2d3a3...	
	New key	Value	Description



Authorization | Headers (2) | Body ● | Pre-request Script | Tests

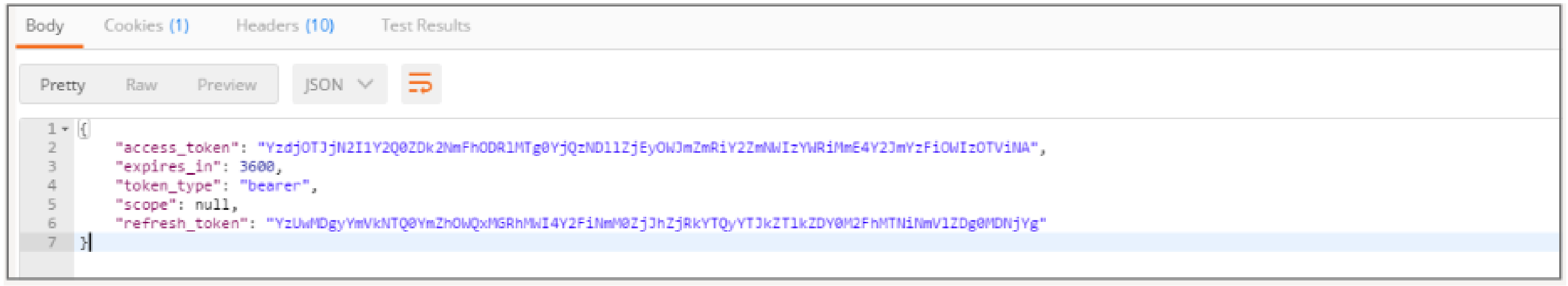
form-data | x-www-form-urlencoded | [✓ raw] | binary | JSON (application/json)

```
{
        "grant_type": "password",
        "username": "test_user@myer.com.au",
        "password": "test1234"
    }
```

MYER MY STORE

API TECHNICAL SPECS

Example (Response)



```
{
    "access_token": "YzdjOTJjN2I1Y2Q0ZDk2NmFhODRlMTg0YjQzNDllZjEyOWJmZmRiY2ZmNWIzYWRiMmE4Y2JmYzFiOWIzOTViNA",
    "expires_in": 3600,
    "token_type": "bearer",
    "scope": null,
    "refresh_token": "YzUwMDgyYmVkNTQ0YmZhOWQxMGRhMWI4Y2FiNmM0ZjJhZjRkYTQyYTJkZTlkZDY0M2FhMTNiNmVlZDg0MDNjYg"
}
```

3. ACCESS TOKEN

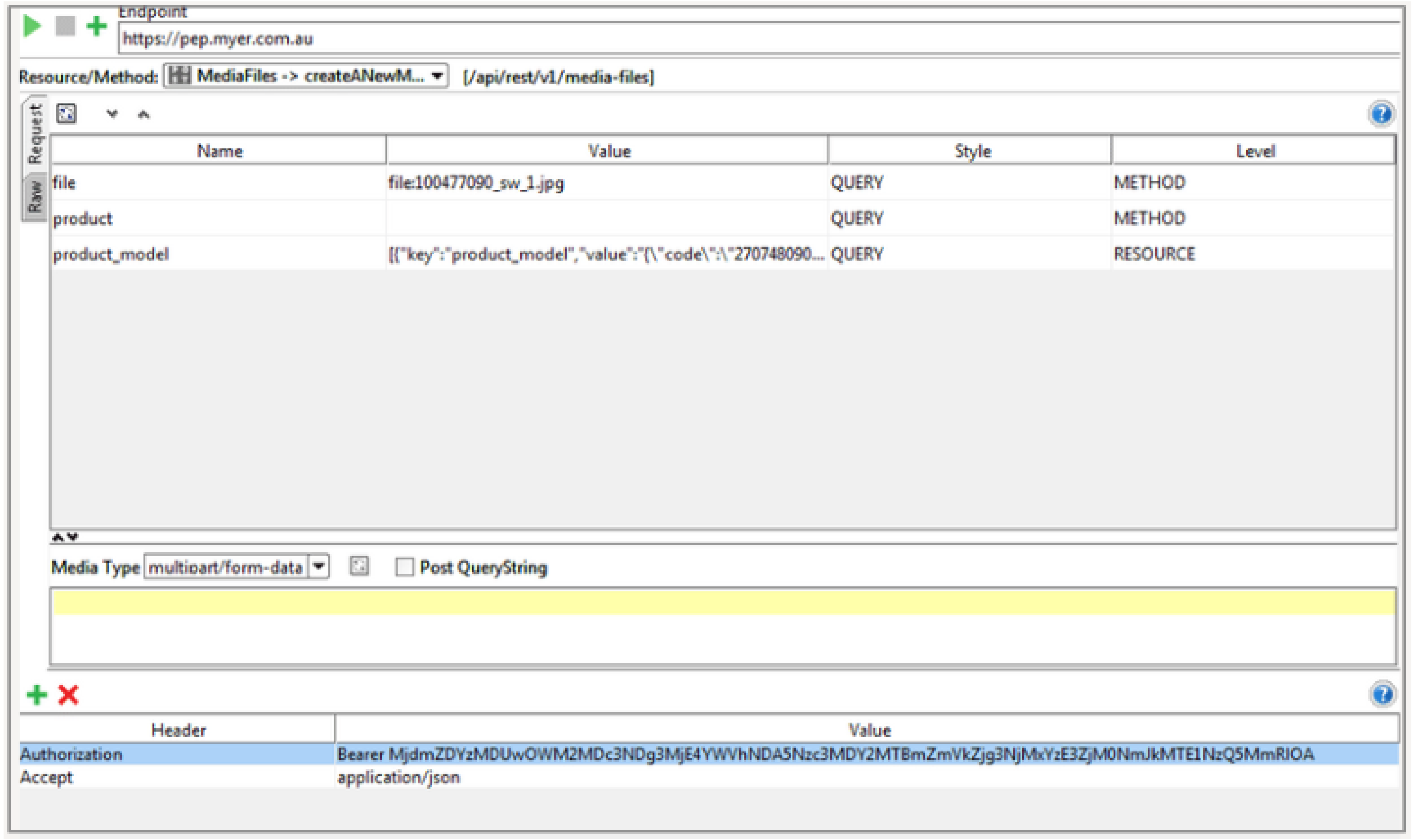
After successful authentication Access_token is returned in the Response. This “access_token” should be prefixed with “Bearer” and used as “Authorization” Header in all the method calls

For Example

Bearer
YzdjOTJjN2I1Y2Q0ZDk2NmFhODRlMTg0YjQzNDllZjEyOWJmZmRiY2ZmNWIzYWRiMmE4Y2JmYzFiOWIzOTViNA

API TECHNICAL SPECS

Example (Request)



Endpoint
https://pep.myer.com.au

Resource/Method: MediaFiles -> createANewM... [/api/rest/v1/media-files]

Name	Value	Style	Level
file	file:100477090_sw_1.jpg	QUERY	METHOD
product		QUERY	METHOD
product_model	[{"key":"product_model","value":"{\"code\":\"270748090...	QUERY	RESOURCE

Media Type multipart/form-data

Post QueryString

Header	Value
Authorization	Bearer MjdmZDYzMDUwOWM2MDc3NDg3MjE4YWVhNDA5Nzc3MDY2MTBmZmVkZjg3NjMxYzE3ZjM0NmJkMTE1NzQ5MmRlOA
Accept	application/json

API TECHNICAL SPECS

4. WORKING WITH PRODUCTS (PRODUCT MODEL)

Following actions can be performed

- Get list of products
- Get a product
- Update list of products
- Update a product

API Reference URL
https://api.akeneo.com/api-reference.html#Productmodel

API TECHNICAL SPECS

Example

Enrich/Update a Product

Endpoint

Headers

Authorization: Generated Access Token

Content-Type: application/json

Body

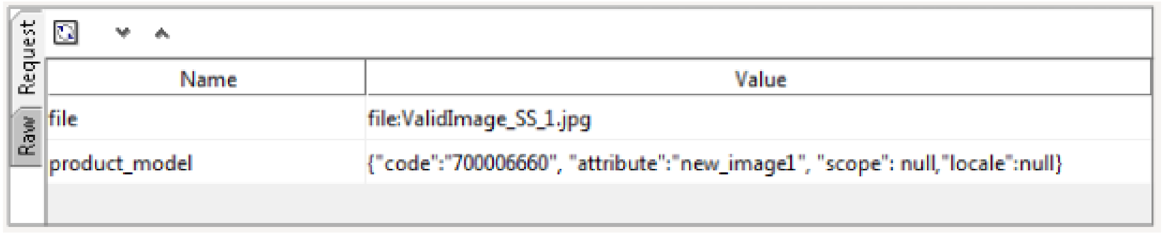
```
{
  "family_variant":"default_variant_size1",
  "values":{
    "copy_status":[
      {
        "locale":null,
        "scope":null,
        "data":"20"
      }
    ],
     "online_name":[
      {
        "locale":"en_AU",
        "scope":"ecommerce",
        "data":"Online name"
      }
     ],
     "supplier_style":[
      {
        "locale":"en_AU",
        "scope":"ecommerce",
        "data":"New Product Model"
      }
     ],
     "online_long_desc":[
      {
        "locale":"en_AU",
        "scope":"ecommerce",
        "data":"Updated Long Desc"
      }
     ],
     "online_short_desc":[
      {
        "locale":"en_AU",
        "scope":"ecommerce",
        "data":"Online11111 Short Desc"
      }
     ],
     "keywords":[
      {
        "locale":"en_AU",
        "scope":"ecommerce",
        "data":null
      }
     ]
   }
}
```

API TECHNICAL SPECS

5. IMAGE UPLOAD

Following actions can be performed Upload image file to a Product

Example Request



Request
Raw

Name	Value
file	file:ValidImage_SS_1.jpg
product_model	{"code":"700006660", "attribute":"new_image1", "scope": null,"locale":null}

API Reference URL
https://api.akeneo.com/api-reference.html#post_media_files

Examples
Image Upload Endpoint: [POST]
Headers Authorization: Generated Access Token Content-Type: multipart/form-data

Body Key: product_model value: {"code":"700000540", "attribute":"new_image1", "scope": null,"locale":null}

Key: file value: file:ValidImage_SS_1.jpg File need to be attached and filename should match filename in value "file:filename.jpg"

API TECHNICAL SPECS

Image Status Update

Endpoint

Headers

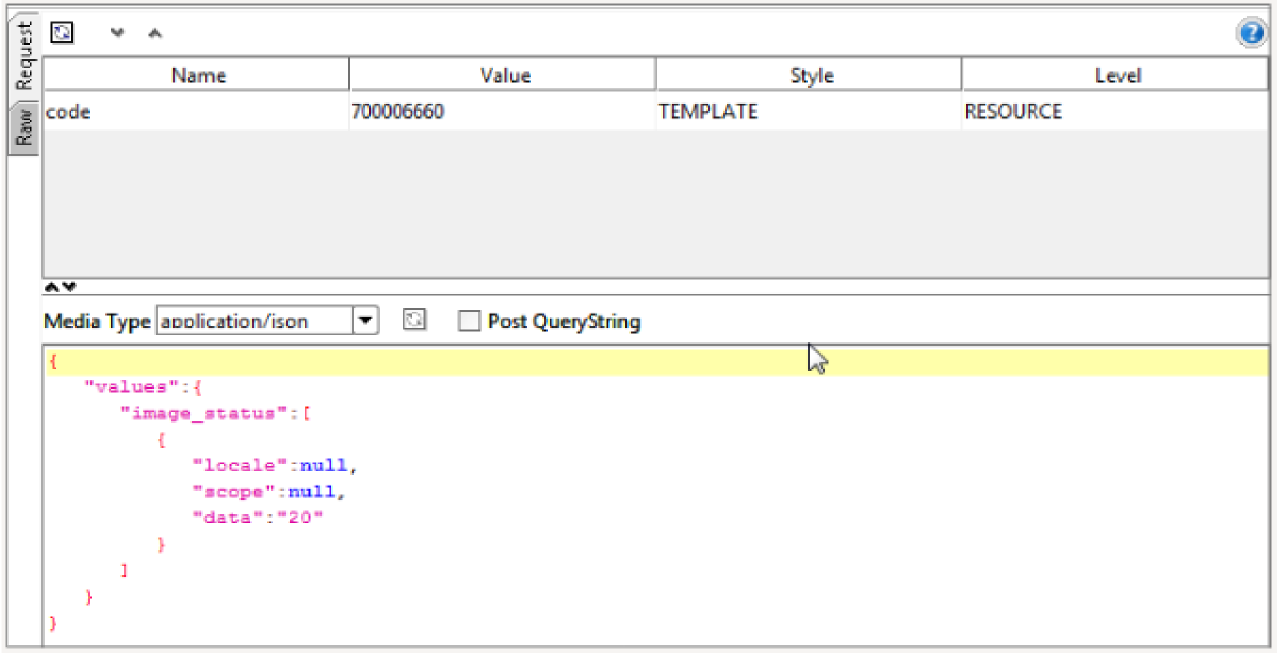
Authorization: Generated Access Token

Content-Type: application/json

Body

```
{
  "values":{
    "image_status":[
      {
        "locale":null,
        "scope":null,
        "data":"20"
      }
    ]
  }
          }
```

Example



Name	Value	Style	Level
code	700006660	TEMPLATE	RESOURCE

```
{
   "values":{
      "image_status":[
         {
            "locale":null,
            "scope":null,
            "data":"20"
         }
      ]
   }
}
```

API TECHNICAL SPECS

Here are the end-points -

*/api/rest/v1/**product-models*** - for Level1 (Parent) => ID(UI) = code(API). *(Copy & Image Enrichment is done at product model level and it gets auto inherited by SKUs)*

*/api/rest/v1/**products*** - for Level2 (SKUs) => ID(UI) = identifier(API). *(This will be used only if promotional prices are to be loaded or any SKU level specific attribute)*