

XMLAPI2

October 26, 2021

1 Eating a Google feed

1.1 Setting up a Shopify private app in Python

```
[1]: %load_ext dotenv
      %dotenv
```

```
[2]: import os
      # Einlesen des API-keys und des Passworts aus einer environment datei
      API_KEY = os.getenv("API_KEY")
      PASSWORD = os.getenv("PASSWORD")
      CREDENTIAL = f'{API_KEY}:{PASSWORD}'

      SHOP = os.getenv("SHOP")
```

```
[3]: import base64
      auth = base64.b64encode(CREDENTIAL.encode('ascii')).decode('ascii')
      headers = {"Accept": "application/json", "Content-Type": "application/json",
      ↪      ↪'Authorization': 'Basic '+auth}
```

```
[4]: API_VERSION = '2021-10'
      base = f'https://{SHOP}/admin/api/{API_VERSION}/'
```

```
[5]: import requests
      import json
      import xmldict
```

1.2 Loading the feed

```
[6]: r = requests.get("https://www.sofaszumhalbenpreis.de/sitemap/googleshoppingfeed.
      ↪xml?r=1632728502")
```

Using xmldict to make a JSON out of the XML

```
[7]: res = xmldict.parse(r.text)
      prod = res['rss']['channel']['item']
```

There are several products within the feed

```
[8]: len(prod)
```

```
[8]: 4869
```

```
[9]: prod[0].keys()
```

```
[9]: odict_keys(['g:id', 'title', 'link', 'g:brand', 'g:price', 'g:description',  
'g:condition', 'g:google_product_category', 'g:product_type', 'g:image_link',  
'g:availability', 'g:quantity', 'g:gtin', 'g:mpn', 'g:shipping',  
'g:custom_label_0', 'g:custom_label_1'])
```

1.3 Create products in Shopify

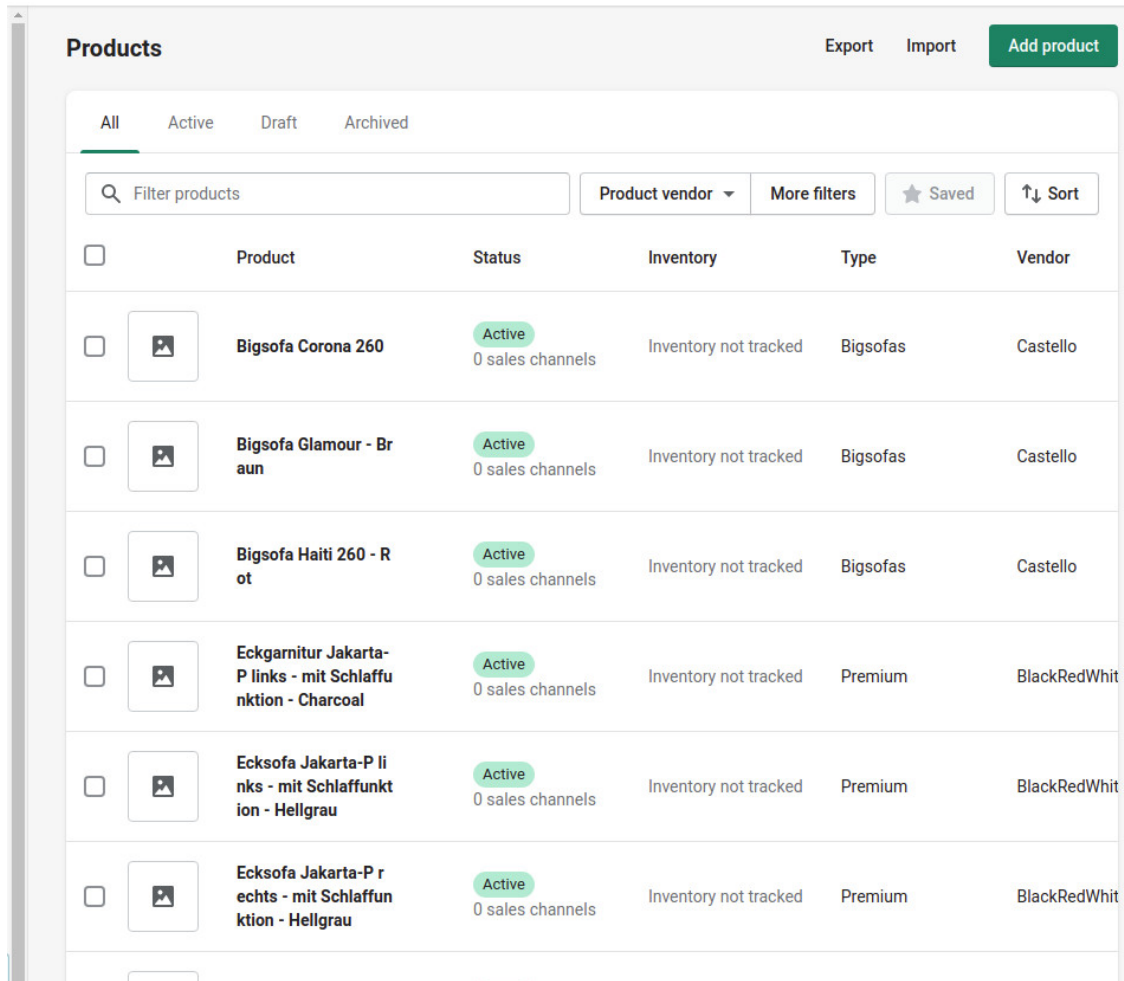
```
[10]: query="""  
mutation productCreate($input: ProductInput!) {  
  productCreate(input: $input) {  
    shop {  
      name  
    }  
    userErrors {  
      field  
      message  
    }  
    product {  
      id  
    }  
  }  
}  
"""
```

Just adding the first one hundred:







```
[11]: pp = []  
for p in prod[:100]:  
    variables={"input":{  
        'title': p['title'],  
        'bodyHtml': p['g:description'][0].replace('\n', ''),  
        'productType': p['g:product_type'],  
        'vendor': p['g:brand']  
    }  
}  
    r = requests.post(base+'graphql.json', json={'query': query, 'variables':  
↪variables}, headers=headers)  
    if r.ok:  
        pp.append(r.json())  
    else:  
        print(r.text)
```

break

Now the products are available within the Shopify shop.



The screenshot shows the Shopify 'Products' management interface. At the top, there are tabs for 'All', 'Active', 'Draft', and 'Archived', with 'All' selected. To the right, there are buttons for 'Export', 'Import', and a green 'Add product' button. Below the tabs is a search bar labeled 'Filter products' and several filter buttons: 'Product vendor', 'More filters', 'Saved', and 'Sort'. The main content is a table with columns for checkboxes, Product, Status, Inventory, Type, and Vendor. The table lists six products, all of which are 'Active' and have '0 sales channels'. The inventory for all products is 'Inventory not tracked'. The product types are 'Bigsofas' and 'Premium', and the vendors are 'Castello' and 'BlackRedWhit'.

<input type="checkbox"/>	Product	Status	Inventory	Type	Vendor
<input type="checkbox"/>	 Bigsofa Corona 260	Active 0 sales channels	Inventory not tracked	Bigsofas	Castello
<input type="checkbox"/>	 Bigsofa Glamour - Braun	Active 0 sales channels	Inventory not tracked	Bigsofas	Castello
<input type="checkbox"/>	 Bigsofa Haiti 260 - Rot	Active 0 sales channels	Inventory not tracked	Bigsofas	Castello
<input type="checkbox"/>	 Eckgarnitur Jakarta-Plinks - mit Schlafunktion - Charcoal	Active 0 sales channels	Inventory not tracked	Premium	BlackRedWhit
<input type="checkbox"/>	 Ecksofa Jakarta-Plinks - mit Schlafunktion - Hellgrau	Active 0 sales channels	Inventory not tracked	Premium	BlackRedWhit
<input type="checkbox"/>	 Ecksofa Jakarta-Pr echts - mit Schlafunktion - Hellgrau	Active 0 sales channels	Inventory not tracked	Premium	BlackRedWhit