

Invoke your API from Developer Portal using an API Key

Lab Objectives

In this lab, you will invoke the secured Fruit Shop API from Developer Portal using its Try functionality. You will act as a self-registered Developer Portal API Consumer. To get access to the secured API, you request for an Application which provides the necessary API Key.

Steps

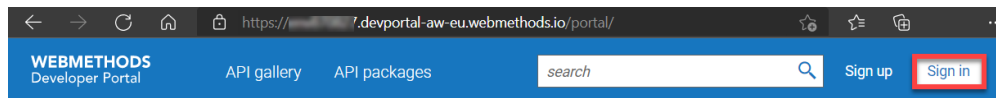
1. Login to your trial Developer Portal in the cloud as self-registered API Consumer acting as a 3rd party developer:

- a. In a browser tab, use the URL to directly access the Developer Portal welcome page:
(https://<your_environment>.<SoftwareAG-Cloud-domain>/portal/)

Note:

You bookmarked this URL in a previous lab of e-learning "Introducing Developer Portal". Only in case you haven't created a Developer Portal API Consumer via self-registration yet, perform steps 2 and 3 of hands-on lab "Discover and Try an API in your Developer Portal" as described in the e-learning "Introducing Developer Portal" first.

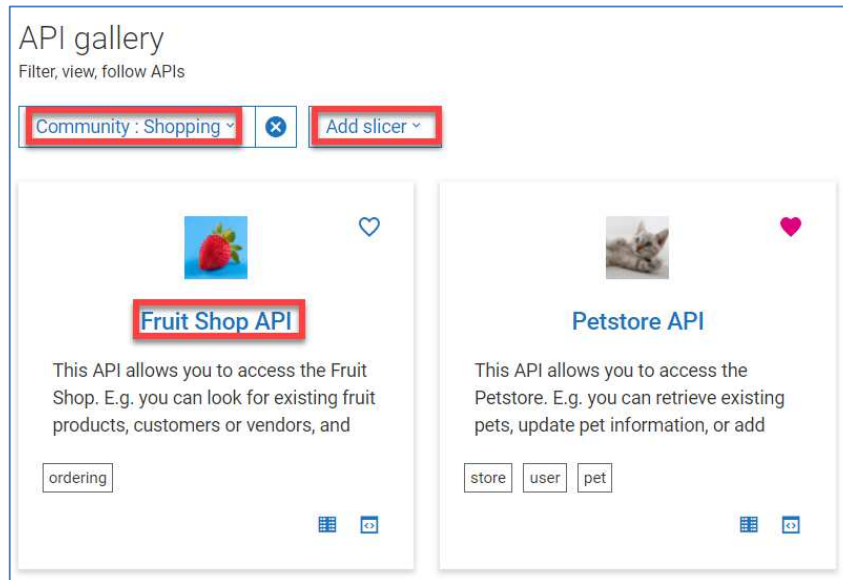
- b. On the welcome page click **Sign in**.



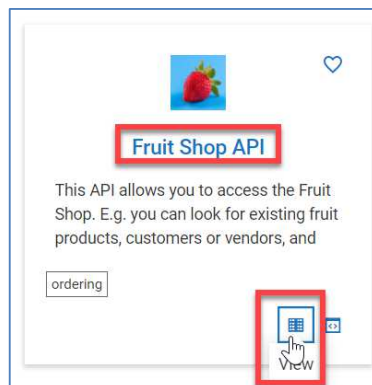
- c. On the sign-in page you can now sign-in to Developer Portal as a Developer Portal API Consumer using the e-mail address and password you defined in hands-on lab "Discover and Try an API in your Developer Portal" of e-learning module "Introducing Developer Portal".

2. Open the **API gallery**.

- Add a slicer for **Community** and select community **Shopping** from the drop-down.
- Ensure the **Fruit Shop API** is still listed for community Shopping.

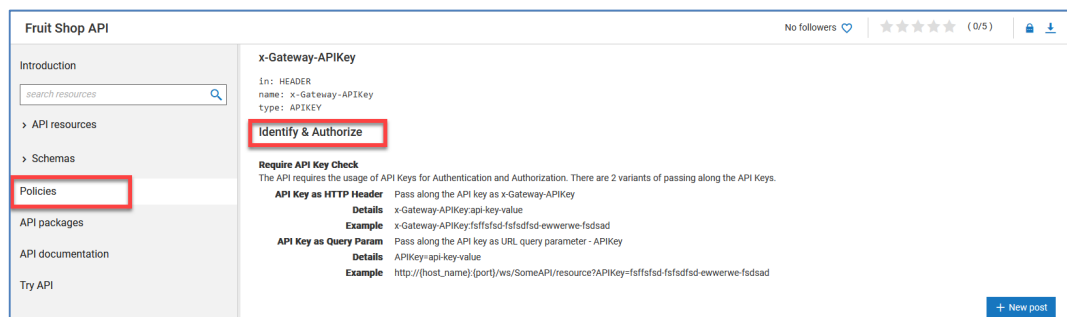


- Click the **View** icon for API **Fruit Shop API** (or click on the API name) to open the details view of the API.



In the left-hand navigation pane, click on **Policies**. The Identify & Authorize policy will be displayed.

Moreover it is mentioned that the API requires the usage of API Keys for Authentication and Authorization.



- d. Click the **Consume** icon to request for a required Application that contains the mandatory API Key as access token.

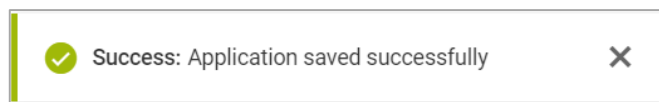


Confirm to request a new application from API Gateway.

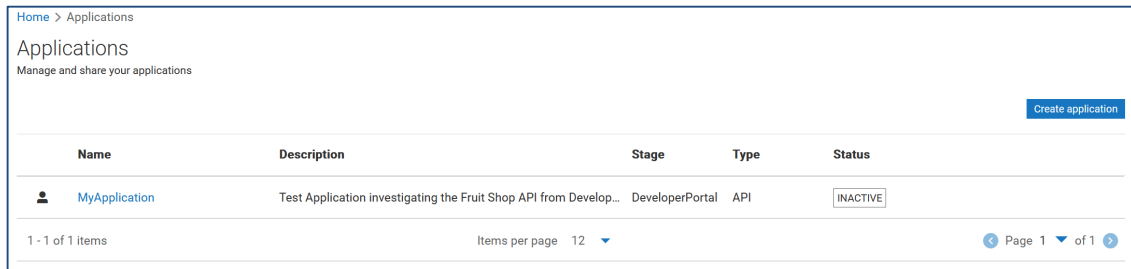


- e. On the Create application panel, leave Stage and the API unchanged, and provide **MyApplication** as Application name as well as a description text of your choice. Eventually click **Save**.

You should get a confirmation like this:



3. Developer Portal will automatically open the Applications page. Your requested Application should be listed here in status **INACTIVE**.



Home > Applications

Applications
Manage and share your applications

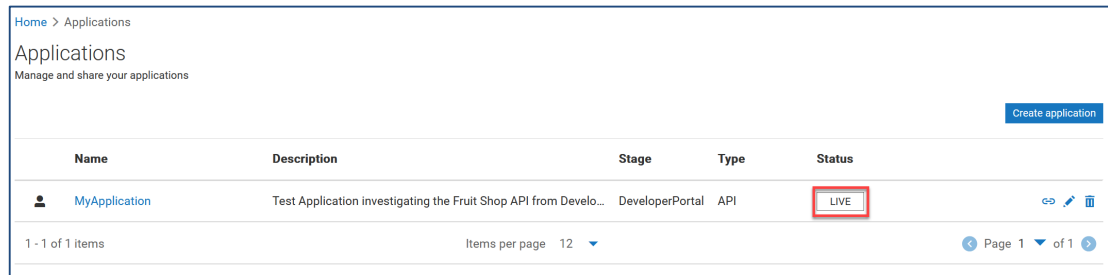
Create application

Name	Description	Stage	Type	Status
MyApplication	Test Application investigating the Fruit Shop API from Develop...	DeveloperPortal	API	INACTIVE

1 - 1 of 1 Items Items per page 12 Page 1 of 1

Note: In case the Applications page is not opened by default, you open the page from the User menu (**Manage applications**).

- a. After a few seconds, refresh the page in your browser. The Application status should have changed to **LIVE**. Open Application **MyApplication** by clicking its name.



Home > Applications

Applications
Manage and share your applications

Create application

Name	Description	Stage	Type	Status
MyApplication	Test Application investigating the Fruit Shop API from Develop...	DeveloperPortal	API	LIVE

1 - 1 of 1 Items Items per page 12 Page 1 of 1

Note: If the Application is still in status INACTIVE, refresh the page after a couple of seconds again. If the status doesn't change at all, double check the credentials you provided in hands-on lab "Secure and Publish your API" at step 3c and re-publish the settings.

- b. As you are the requester/owner of the Application MyApplication, you are allowed to see the generated contained API key value which can be used as a valid access token. Un-hide the value of the API key and copy the value into the clipboard.

MyApplication

Create and manage applications

Test Application Investigating the Fruit Shop API from Developer Portal.

Access tokens

API key

API key

Expiry date

OAuth

Client id

Client secret

Token validity (in milliseconds)

Refresh count

Authorization URL

Access token URL

Scopes

Redirect URL(s)

JWT

JWT default claims set

app_id

APIs

Name	Summary	Version
Fruit Shop API	This API allows you to access the Fruit Shop. E.g. you can look for existing fruit products, customers or ven...	1.0.0

- c. At the very bottom of the Application details page, the associated API(s) are listed for MyApplication. Select the **Fruit Shop API** to switch back to the API's details page.

APIs

Name	Summary	Version
Fruit Shop API	This API allows you to access the Fruit Shop. E.g. you can look for existing fruit products, customers or ven...	1.0.0

4. On the Fruit Shop API details page, select **Try API** from the left-hand navigation bar.
 - a. On the Try API page, expand API resource **/products/** in the left-hand navigation bar and select method **getProducts**.
 - b. Open tab **Authorization**. If not already pre-selected, select **MyApplication** as Application and set Authorization type to **API key**.
 - c. Switch to tab **Headers**. Ensure key **Content-Type** is set to **application/json** and a key named **x-Gateway-APIKey** exists with the same API key value still contained in your clipboard.

The screenshot shows the 'Fruit Shop API' interface. On the left, the 'API resources' list has '/products/' selected, with 'getProducts' highlighted. The main area shows the 'Headers' tab for the GET method at the endpoint 'http://env870827.apigw-aw-eu.webmethods.io/gateway/Fruit%20Shop%20API/1.0.0/products/'. Two headers are configured: 'Content-Type' with value 'application/json' and 'x-Gateway-APIKey' with value 'db55413f-95af-4a14-9d8c-c8e53cc7a3f4'. A 'Send' button is visible in the top right.

- d. Click **Send** to get a list of all available fruits in json format returned in the response body.

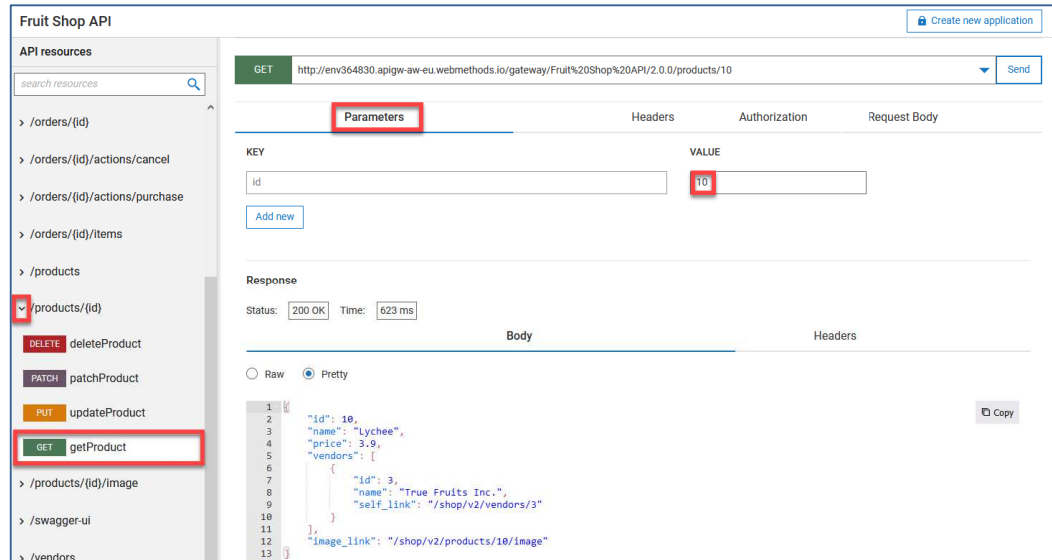
The screenshot shows the API response body in 'Pretty' format. The JSON structure includes a 'meta' object with 'count', 'start', 'limit', and 'next_link' properties, and a 'products' array containing four product objects. Each product object has 'id', 'name', and 'self_link' properties.

```

1  {
2    "meta": {
3      "count": 86,
4      "start": 1,
5      "limit": 10,
6      "next_link": "/shop/v2/products?start=11&limit=10"
7    },
8    "products": [
9      {
10       "id": 113,
11       "name": "Mango fresh",
12       "self_link": "/shop/v2/products/113"
13     },
14     {
15       "id": 111,
16       "name": "Salad",
17       "self_link": "/shop/v2/products/111"
18     },
19     {
20       "id": 110,
21       "name": "Mango fresh",
22       "self_link": "/shop/v2/products/110"
23     },
24     {
25       "id": 108,
26       "name": "Salad",
27       "self_link": "/shop/v2/products/108"
28     },
29     {
30       "id": 107,
31       "name": "Mango fresh",
32       "self_link": "/shop/v2/products/107"
33     }
34   ]
35 }

```

- e. Expand API resource **/products/{id}** in the left-hand navigation bar and select method **getProduct**.
- f. On tab **Parameters**, assign value **10** to key **id**.
- g. Click **Send** again. Detailed product data of product 10 (Lychee) will be returned.



The screenshot shows the 'Fruit Shop API' interface. On the left, the 'API resources' sidebar lists various endpoints. The endpoint **/products/{id}** is selected, and the **getProduct** method is highlighted with a red box. The main panel shows the **Parameters** tab selected, with a table where the key **id** is assigned the value **10**. The **Send** button is visible in the top right. Below the parameters, the **Response** section shows a status of **200 OK** and a time of **623 ms**. The response body is displayed in a pretty-printed JSON format, showing details for product 10 (Lychee), including its price and vendor information.

```

{
  "id": 10,
  "name": "Lychee",
  "price": 3.9,
  "vendors": [
    {
      "id": 3,
      "name": "True Fruits Inc.",
      "self_link": "/shop/v2/vendors/3"
    }
  ],
  "image_link": "/shop/v2/products/10/image"
}
    
```