App Modernization Labs

Azure API Management

Author: Divi Mishra, Francis Nazareth

Cloud Solution Architect, Azure App Innovation, Microsoft Qatar

divimishra@microsoft.com | +974 50219105

fnazaret@microsoft.com | +974 33134296

Introduction to the Lab Document

Objective

This workshop lab is intended to materialize on theoretical Azure API Management learnings to have hands-on experience with end-to-end Azure API Management tools. Through this workshop lab, you will have a basic yet broad understanding of how to realize value from the different offerings within and beyond Azure API Management.

Intended Audience

The intended audience for this workshop lab includes, but is not limited to: development team, application managers, enterprise architects, and technical managers. The difficulty level of this workshop is beginners.

Duration

This workshop lab followed step-by-step will take approximately 3 hours to complete.

Pre-requisites

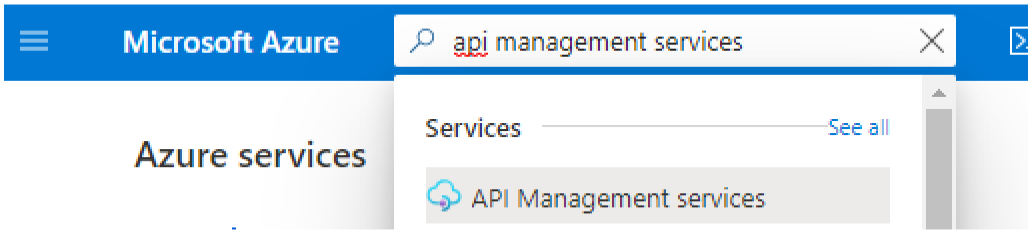
1. Access to an active Azure subscription
2. Contributor Role in a resource group to be able to deploy an api management instance

TABLE OF CONTENTS

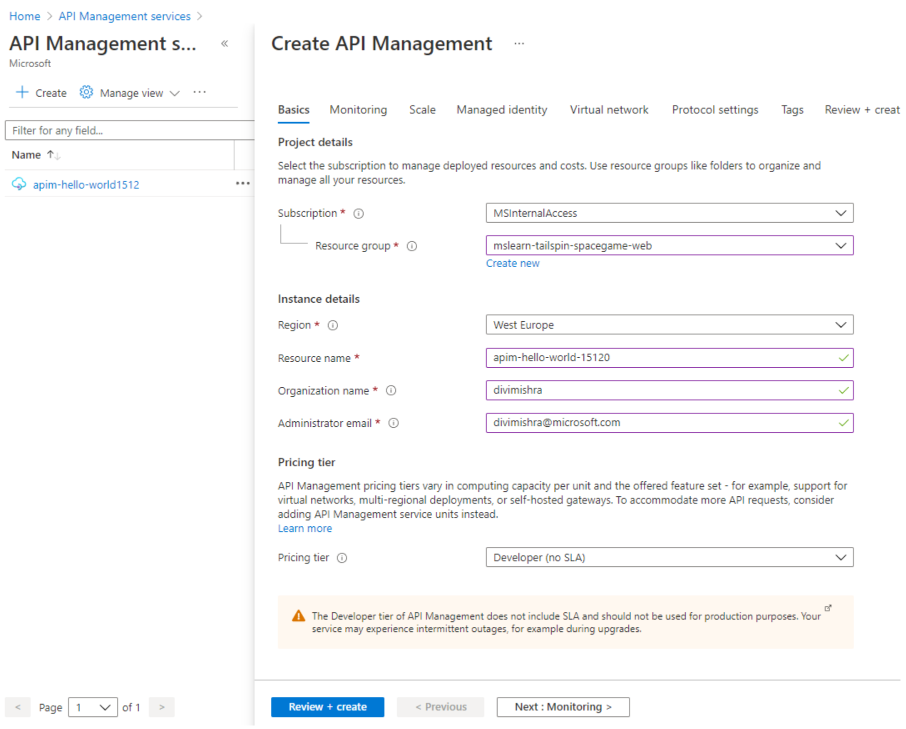
Pre-requisites: Azure API Management

Create an APIM instance

1. Log into Azure Portal.
2. Search for **API Management Services** at the search bar on top



3. Click on **+ Create**
4. Under **Basics:**
   a. Select a relevant subscription, resource group, region.
   b. Enter a unique resource name.
   c. Enter a relevant organization name and administrator email address for notification purposes.
   d. Select **Developer (no SLA)** as your Pricing tier.



Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	MSInternalAccess
Resource group *	mslearn-tailspin-spacegame-web

Create new

Instance details

Region *	West Europe
Resource name *	apim-hello-world-15120
Organization name *	divimishra
Administrator email *	divimishra@microsoft.com

Pricing tier

API Management pricing tiers vary in computing capacity per unit and the offered feature set - for example, support for virtual networks, multi-regional deployments, or self-hosted gateways. To accommodate more API requests, consider adding API Management service units instead.
Learn more

Pricing tier	Developer (no SLA)

The Developer tier of API Management does not include SLA and should not be used for production purposes. Your service may experience intermittent outages, for example during upgrades.

5. Please go through the other tabs to view the options. Accept defaults for every other tabs.
6. Click on **Review + Create.**
7. Then, click on **Create.**

NOTE:

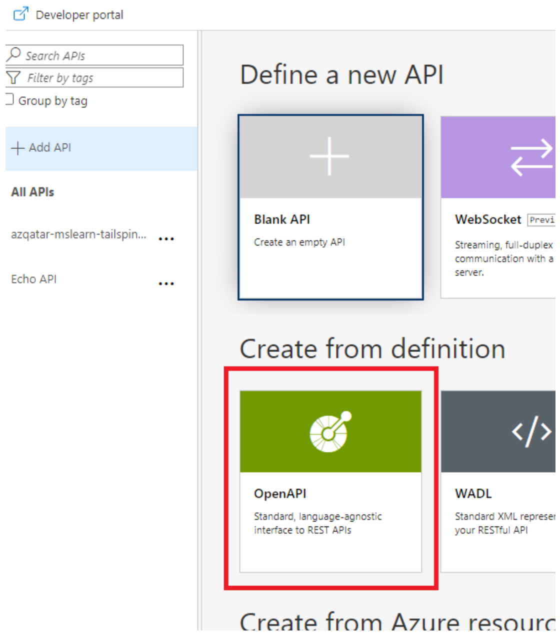
Creating an APIM instance can take more than 30 minutes.

Lab 01: Create your first API

Import and publish your first API

In this lab, we will import an API specification (OpenAPI swagger definition) from the link https://conferenceapi.azurewebsites.net?format=json to create API end points.

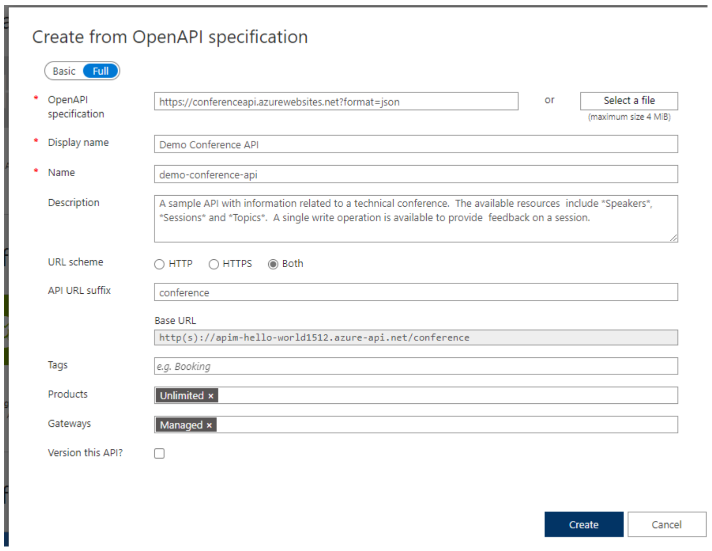
1. Go to the APIM resource deployed.
2. Under **APIs,** click on **APIs**.
3. In the create from definition section, Select **OpenAPI.**



4. In the "Create from Open API specification box", Click on **Full** to get the full dialog.

   Please provide the following values:

   a. OpenAPI Sepcification https://conferenceapi.azurewebsites.net?format=json
   b. API URL Suffix: **conference**
   c. Products: **Unlimited**
   d. Leave the rest as defaults



Create from OpenAPI specification

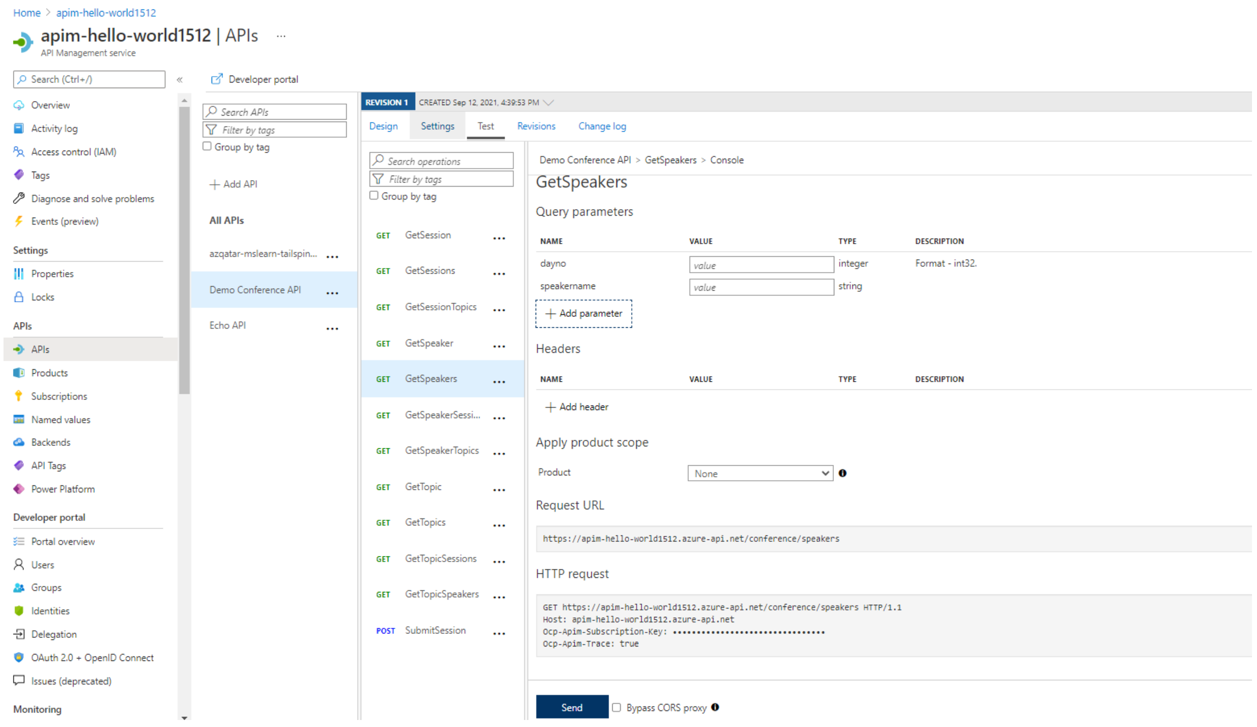
Basic | Full

* OpenAPI specification	https://conferenceapi.azurewebsites.net?format=json
* Display name	Demo Conference API
* Name	demo-conference-api
Description	A sample API with information related to a technical conference. The available resources include "Speakers", "Sessions" and "Topics". A single write operation is available to provide feedback on a session.
URL scheme	○ HTTP ○ HTTPS [✓ Both]
API URL suffix	conference
Base URL	http(s)://apim-hello-world1512.azure-api.net/conference
Tags	e.g. Booking
Products	Unlimited ×
Gateways	Managed ×
Version this API?	☐

5. Click on **Create.**

Test the imported API

1. Click on **Demo Conference API.**
2. Go to the **Test** tab.
3. Select **GetSpeakers.**
4. Note that they Query parameters were obtained from the Open API specification. Note that the Ocp-Apim-Subscription-Key header was also automatically filled in.



GetSpeakers

Query parameters

NAME	VALUE	TYPE	DESCRIPTION
dayno	value	integer	Format - int32.
speakername	value	string	

Headers

NAME	VALUE	TYPE	DESCRIPTION

Apply product scope

Product: None

Request URL

```
https://apim-hello-world1512.azure-api.net/conference/speakers
```

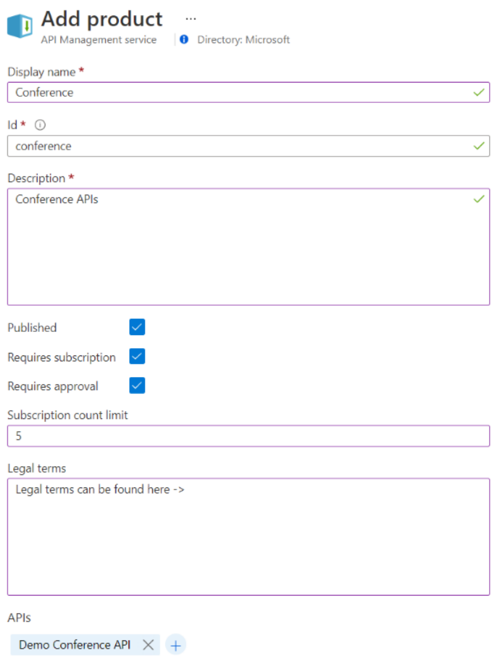
HTTP request

```
GET https://apim-hello-world1512.azure-api.net/conference/speakers HTTP/1.1
Host: apim-hello-world1512.azure-api.net
Ocp-Apim-Subscription-Key: ••••••••••••••••••••••••••••••••
Ocp-Apim-Trace: true
```

5. Click **Send.** You will receive the 200 OK response for successful testing.

Lab 02: Create a Product

1. Click on products on the left-side ribbon, and click on +Add
2. Provide a name for the product (e.g.: Conference)
3. Provide a description for the product
4. Click on “Published” (to publish this product to portal).
5. Click on “Requires subscription”
6. Click on “Requires approval”
7. Set the subscription count limit to a lower value (such as 5)



Display name * Conference

Id * conference

Description * Conference APIs

Published ✓

Requires subscription ✓

Requires approval ✓

Subscription count limit 5

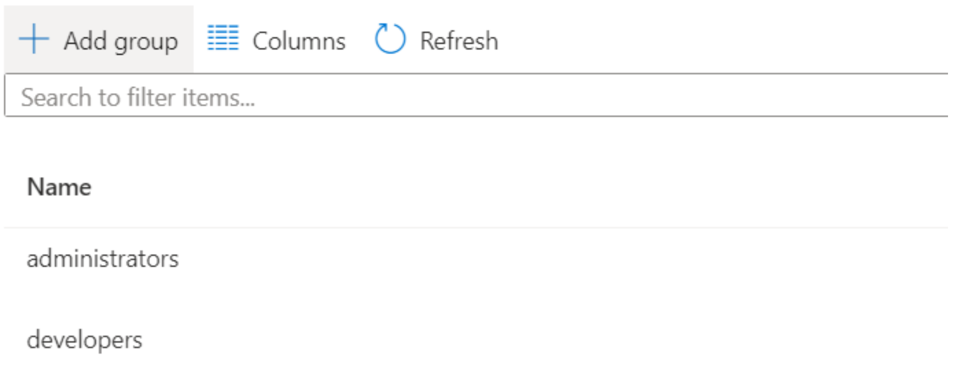
Legal terms Legal terms can be found here ->

APIs Demo Conference API

8. Click “Create”.
9. Once the product is created, click on the product name (or from the left ribbon -> products -> product name).

Add a user group to the product.

10. Click on “Access Control”.
11. Click on “Add Group” and add the built-in group Developers. (These are the signed-in users to the developer portal).

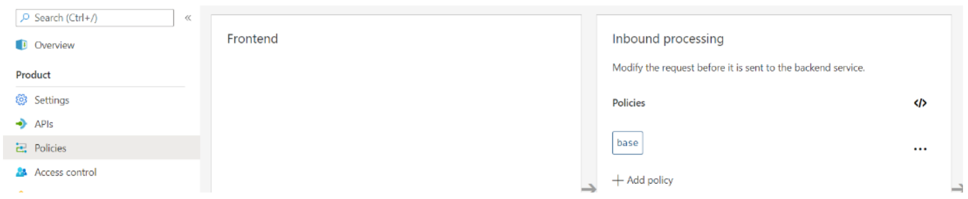


Name
administrators
developers

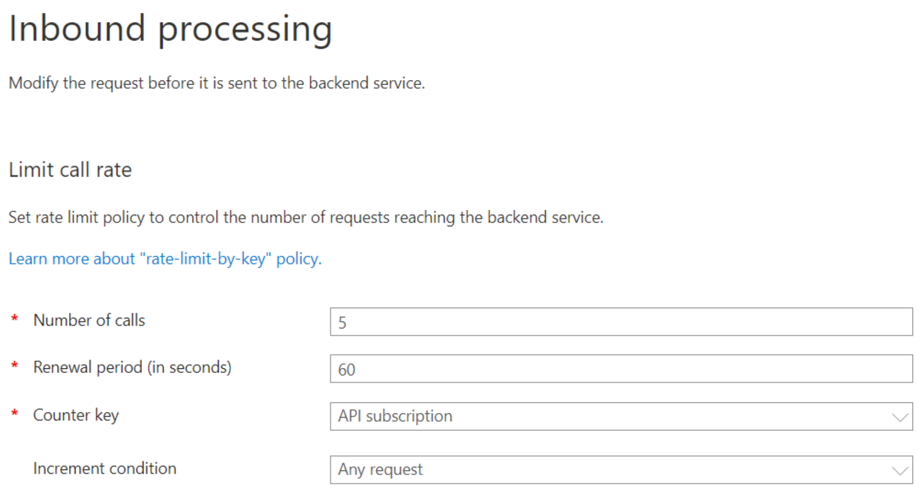
Add a rate limiting policy to the product.

Let us limit the rate of API invocations to 5 requests per minute.

12. While within the product -> product name view, click on “Policies”.
13. In the in-bound processing section, click on “+ Add Policy”.



14. Select the policy “Limit Call Rate”.
15. Select the values as follows:
    a. Number of calls: 5
    b. Renewal period (in seconds): 60


Inbound processing

Modify the request before it is sent to the backend service.

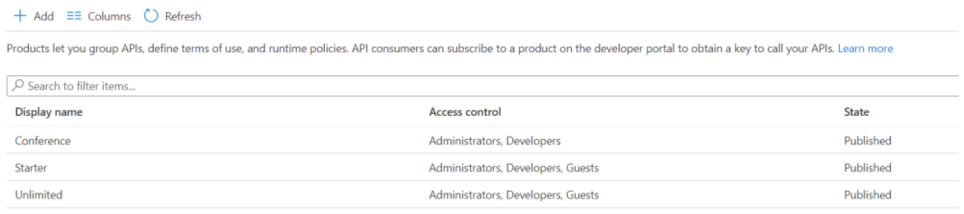
Limit call rate

Set rate limit policy to control the number of requests reaching the backend service.

Learn more about "rate-limit-by-key" policy.

* Number of calls	5
* Renewal period (in seconds)	60
* Counter key	API subscription
Increment condition	Any request

16. Click Save
17. The final view of products will be similar to the following diagram:



Products let you group APIs, define terms of use, and runtime policies. API consumers can subscribe to a product on the developer portal to obtain a key to call your APIs. Learn more

Search to filter items...

Display name	Access control	State
Conference	Administrators, Developers	Published
Starter	Administrators, Developers, Guests	Published
Unlimited	Administrators, Developers, Guests	Published

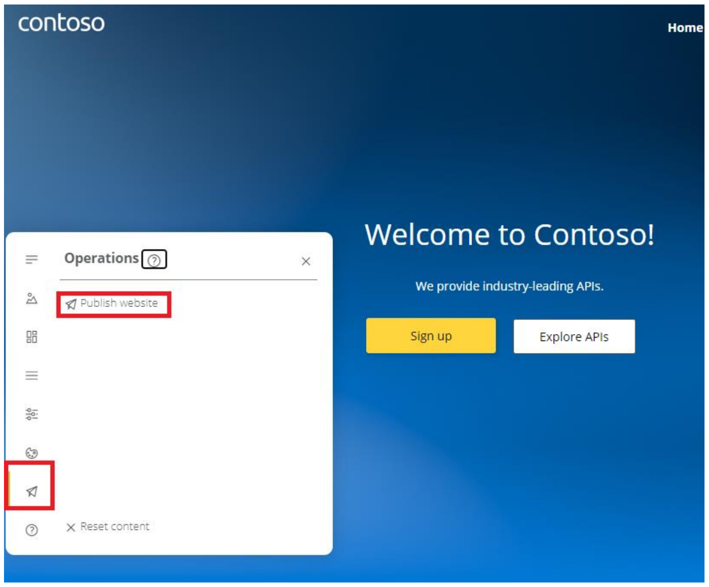
18. (Optional): Remove the "Demo Conference API" from the "Unlimited" product.

Lab 03: Developer Portal

Create your Developer Portal

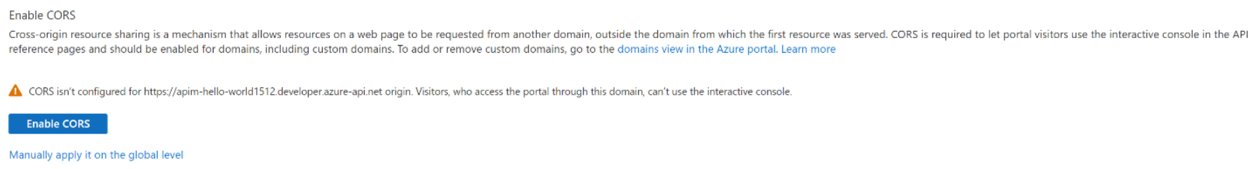
1. Navigate back to the Azure Portal and your API Management instance.
2. Click on **Developer Portal** link at the top of the screen. Open it in a new tab.
3. Wait for the portal website to be created.
   Note: The Developer portal is based on Paperbits framework. This allows drag-and-drop customization. You can also have custom widgets.

4. Try to change the message "Welcome to Contoso!" (double-click on the text to edit)
5. Try to change the message "We provide industry-leading APIs".
6. Try to change the background with another picture. (double-click on the area below sign-up).
7. Try to change the logo with another logo image (if you have a small logo jpg / gif file).
8. Click on the paper airplane icon and click on **Publish website (**To click on buttons and visit their hyperlinks, do **Ctrl + Click.)**



Enable CORS for the Developer Portal

1. Navigate back to the Azure Portal and your API Management instance.
2. Under **Developer Portal** section, go to **Portal Overview.**
3. Click on **Enable CORS.** Click on **Yes.**



Enable CORS

Cross-origin resource sharing is a mechanism that allows resources on a web page to be requested from another domain, outside the domain from which the first resource was served. CORS is required to let portal visitors use the interactive console in the API reference pages and should be enabled for domains, including custom domains. To add or remove custom domains, go to the domains view in the Azure portal. Learn more

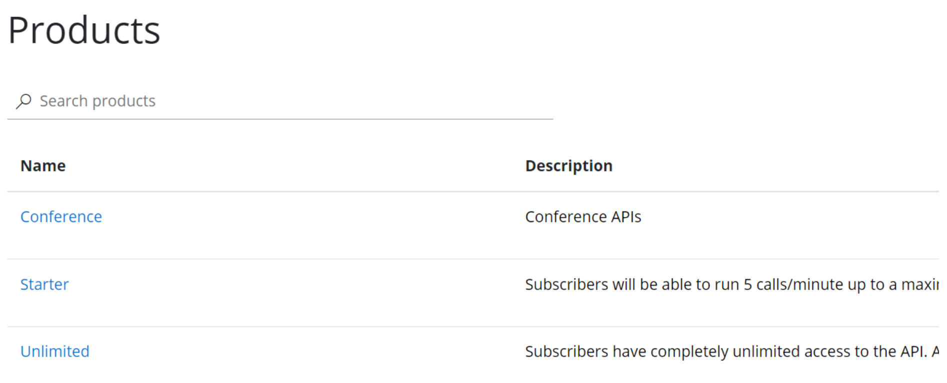
CORS isn't configured for https://apim-hello-world1512.developer.azure-api.net origin. Visitors, who access the portal through this domain, can't use the interactive console.

Register to the Developer Portal.

9. Open the developer portal in a different browser (or in an incognito / in-private window in the same browser).
10. Click on "Products" to view the products. You should be able to see the "Starter" and "Unlimited" product, but not the product you created (since you have not logged in, only products with visibility for the guest group in access control will be visible).
11. Click on "Sign-up"
12. Fill in your personal email details, type in the CAPTCHA, and sign-up.
13. Check your email to verify the API management sign-up process.
14. Once you have completed the verification, click on "Sign-in" on the Developer Portal to login to the portal.

Test APIs through Developer Portal

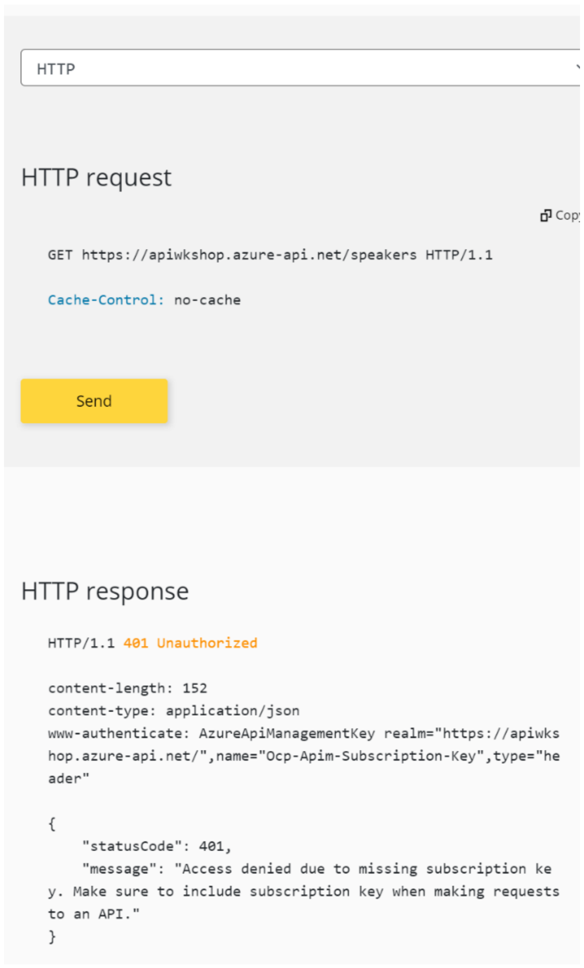
15. Click on Products
16. You should be able to see the product you created now.



Products

Name	Description
Conference	Conference APIs
Starter	Subscribers will be able to run 5 calls/minute up to a maxi
Unlimited	Subscribers have completely unlimited access to the API. A

17. Click on the product name.
18. In the "APIs in this product" section, Click on the "Demo Conference API"
19. Click on the "Get Speakers" and click "Try it".
20. Click "Send".
21. You should get an unauthorized message (we need to subscribe to the product to invoke APIs).



HTTP request

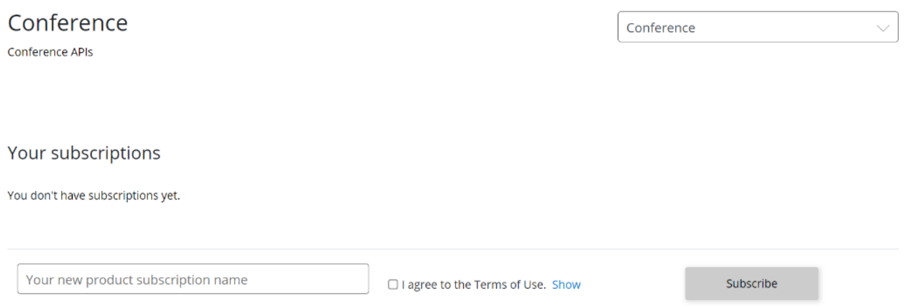
```
GET https://apiwkshop.azure-api.net/speakers HTTP/1.1

Cache-Control: no-cache
```

HTTP response

```
HTTP/1.1 401 Unauthorized

content-length: 152
content-type: application/json
www-authenticate: AzureApiManagementKey realm="https://apiwks
hop.azure-api.net/",name="Ocp-Apim-Subscription-Key",type="he
ader"

{
    "statusCode": 401,
    "message": "Access denied due to missing subscription ke
y. Make sure to include subscription key when making requests
to an API."
}
```

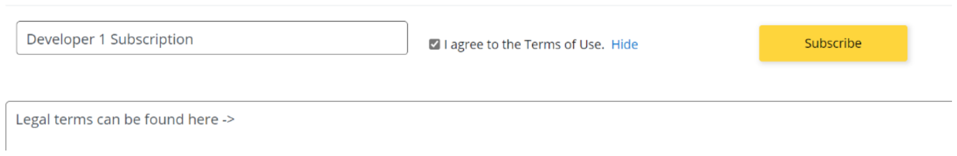
Subscribe to a product

22. Go back to the products -> your product view


Conference
Conference APIs
Conference
Your subscriptions
Your new product subscription name
I agree to the Terms of Use. Show
Subscribe

You don't have subscriptions yet.

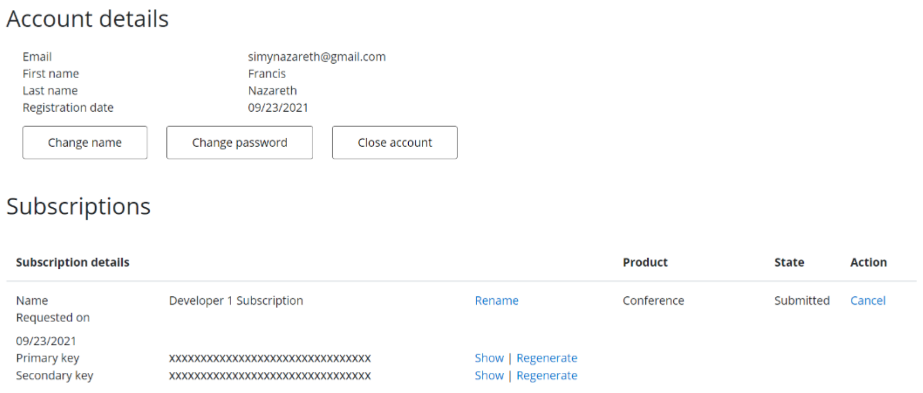
23. In the Your subscriptions section, type in a subscription name (for example, Developer 1 subscription).
24. Click "Show" to view the terms of use, click on the tick mark to accept the terms of use.


Developer 1 Subscription

☑ I agree to the Terms of Use. Hide

Legal terms can be found here ->

25. Click on “Subscribe”.
26. You will see that your subscription request is in the “Submitted” state. (Please remember that when we created the product, we specified that the product subscription requires approval).


Account details

Email	simynazareth@gmail.com
First name	Francis
Last name	Nazareth
Registration date	09/23/2021

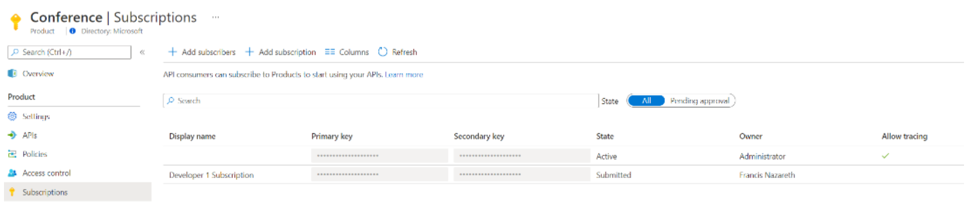
Subscriptions

Subscription details			Product	State	Action
Name	Developer 1 Subscription	Rename	Conference	Submitted	Cancel
Requested on 09/23/2021					
Primary key	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	Show | Regenerate			
Secondary key	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	Show | Regenerate			

27. Click on “Show” to view the value of Primary key and Secondary key. (These headers need to be included in the calls to the API).

Approve an API subscription request.

28. In the initial browser window, go back to Azure API management.
29. From the left ribbon, select “Products” -> <Your product name>
30. Click on “Subscriptions”.


Conference | Subscriptions

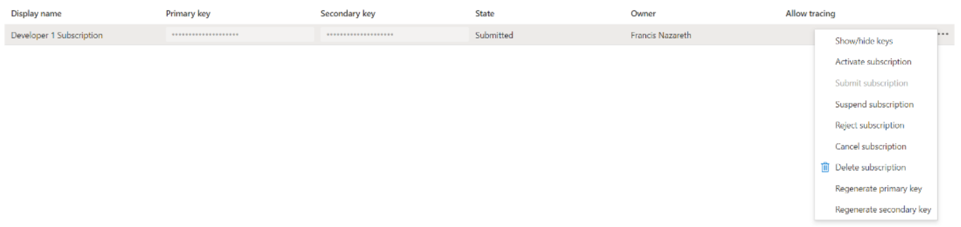
Product | Directory: Microsoft

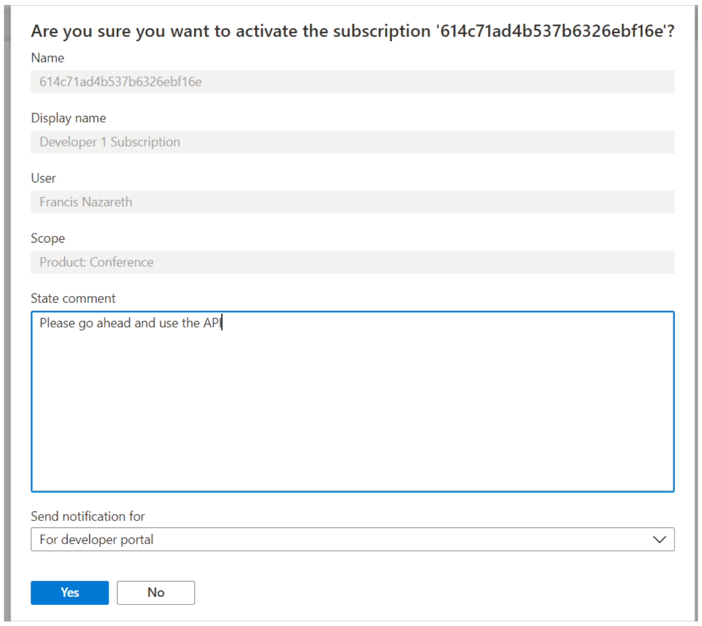
API consumers can subscribe to Products to start using your APIs. Learn more

State: All | Pending approval

Display name	Primary key	Secondary key	State	Owner	Allow tracing
	••••••••••••••••••••	••••••••••••••••••••	Active	Administrator	✓
Developer 1 Subscription	••••••••••••••••••••	••••••••••••••••••••	Submitted	Francis Nazareth	

31. (Optional) Toggle the state to Pending Approval.
32. Click on the “...” symbol on the right side, and from the menu, select “Activate subscription”.


Display name	Primary key	Secondary key	State	Owner	Allow tracing
Developer 1 Subscription	••••••••••••••••••••	••••••••••••••••••••	Submitted	Francis Nazareth	


Are you sure you want to activate the subscription '614c71ad4b537b6326ebf16e'?

Name: 614c71ad4b537b6326ebf16e

Display name: Developer 1 Subscription

User: Francis Nazareth

Scope: Product: Conference

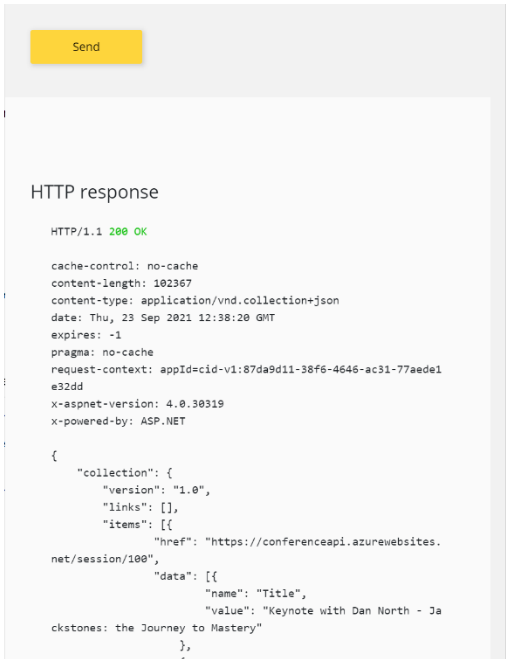
State comment: Please go ahead and use the API

Send notification for: For developer portal

33. Add any comment and click "Yes".
34. You will get an email on your personal email that the subscription is activated.
35. Go back to the developer portal browser window and refresh the user profile page. The subscription status should now be in "Active".

Test the API and Rate Limiting Features.

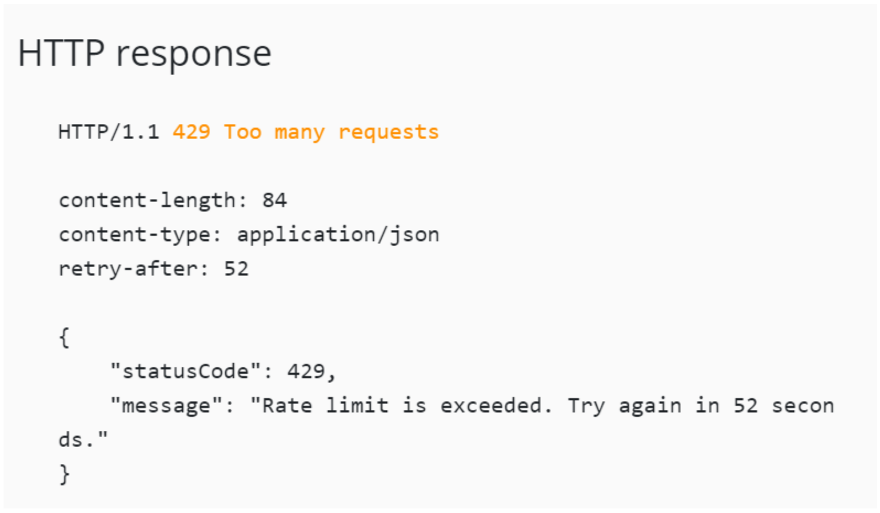
36. Go back to the "Product" menu and select the product you created.
37. Select the "Demo Conference API".
38. Select "Get Sessions" again and click on "Try it".
39. In the subscription key field, you will see that the primary / secondary subscription keys can be selected. Leave it to default (so that the primary subscription key is selected).
40. Click "Send" to invoke the API.
41. You should see the response for API call with 200/OK.



HTTP response

```
HTTP/1.1 200 OK

cache-control: no-cache
content-length: 102367
content-type: application/vnd.collection+json
date: Thu, 23 Sep 2021 12:38:20 GMT
expires: -1
pragma: no-cache
request-context: appId=cid-v1:87da9d11-38f6-4646-ac31-77aede1
e32dd
x-aspnet-version: 4.0.30319
x-powered-by: ASP.NET

{
    "collection": {
        "version": "1.0",
        "links": [],
        "items": [{
                "href": "https://conferenceapi.azurewebsites.
net/session/100",
                "data": [{
                        "name": "Title",
                        "value": "Keynote with Dan North - Ja
ckstones: the Journey to Mastery"
                    },
```

42. Invoke the API multiple times (more than 5 times in a minute). We limited the number of requests per minute to 5 in when we defined the product. You should see HTTP 429 with too many requests as the response.



HTTP response

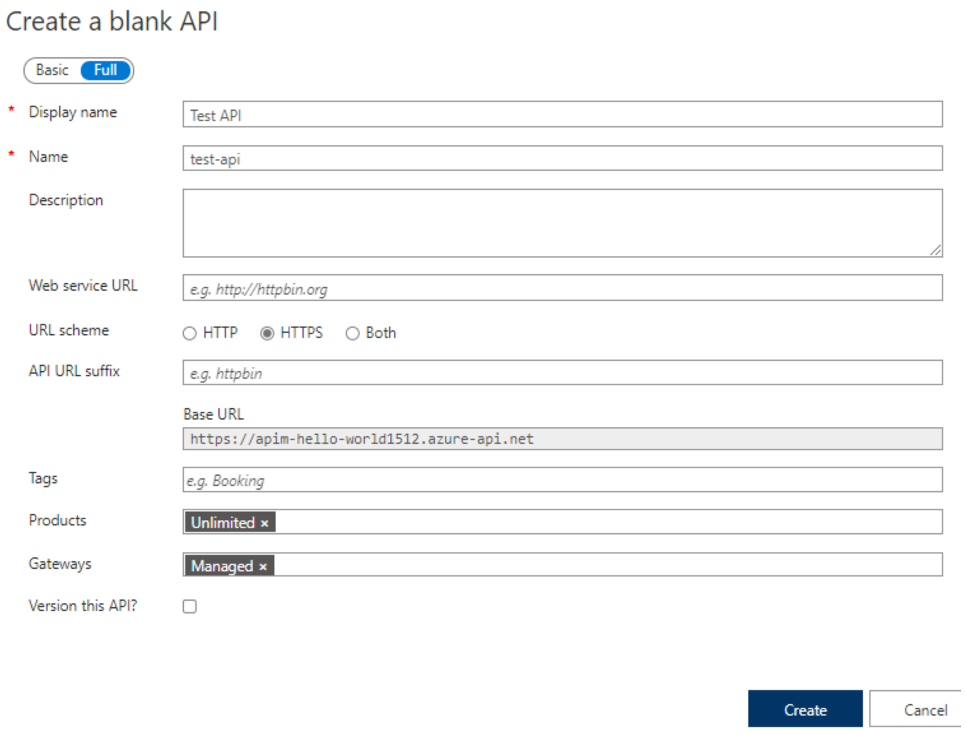
```
HTTP/1.1 429 Too many requests

content-length: 84
content-type: application/json
retry-after: 52

{
    "statusCode": 429,
    "message": "Rate limit is exceeded. Try again in 52 secon
ds."
}
```

Lab 04: Mock an API Response

Create a blank Test API

1. Navigate back to **APIs.**
2. Click on **Blank API.** Click on **Full** for the full dialog.
3. Enter the following details:
    a. **Test API** for the display name.
    b. **Unlimited** for the product
    c. For the API URL prefix, add *mock* (or any other prefix)
    d. Leave the rest as their default values
4. Select **Create**

Note that this API has no backend.



Create a blank API

Basic [✓ Full]

* Display name	Test API
* Name	test-api
Description	
Web service URL	*e.g. http://httpbin.org*
URL scheme	HTTP [✓ HTTPS] Both
API URL suffix	*e.g. httpbin*
Base URL	https://apim-hello-world1512.azure-api.net
Tags	*e.g. Booking*
Products	Unlimited ×
Gateways	Managed ×
Version this API?	☐

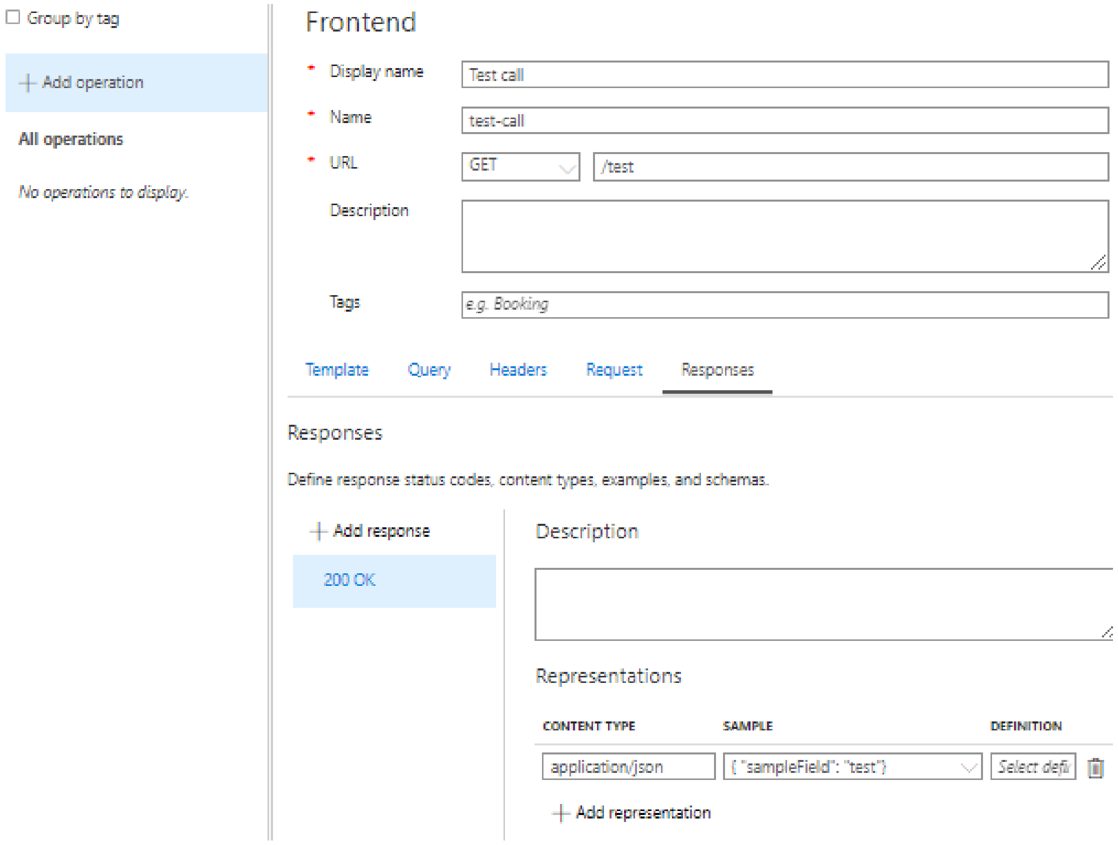
Create a mock API response

1. Select the Test API that was just created.
2. Click **+ Add operation**
3. Enter the following details:
    a. Display name: **Test Op**
    b. URL: **GET** /test
4. Under the **Responses** tab, click on **+ Add response**
5. Select **200 OK**
6. Select **+ Add Representation**
7. Enter the following details:
    a. CONTENT TYPE: Type **application/json** and select it

b. SAMPLE: Type `{"sampleField": "test"}`

c.

8. Select **Save**



Group by tag

Add operation

All operations

No operations to display.

Frontend

Display name: Test call

Name: test-call

URL: GET /test

Description:

Tags: e.g. Booking

Responses

Define response status codes, content types, examples, and schemas.

Add response

200 OK

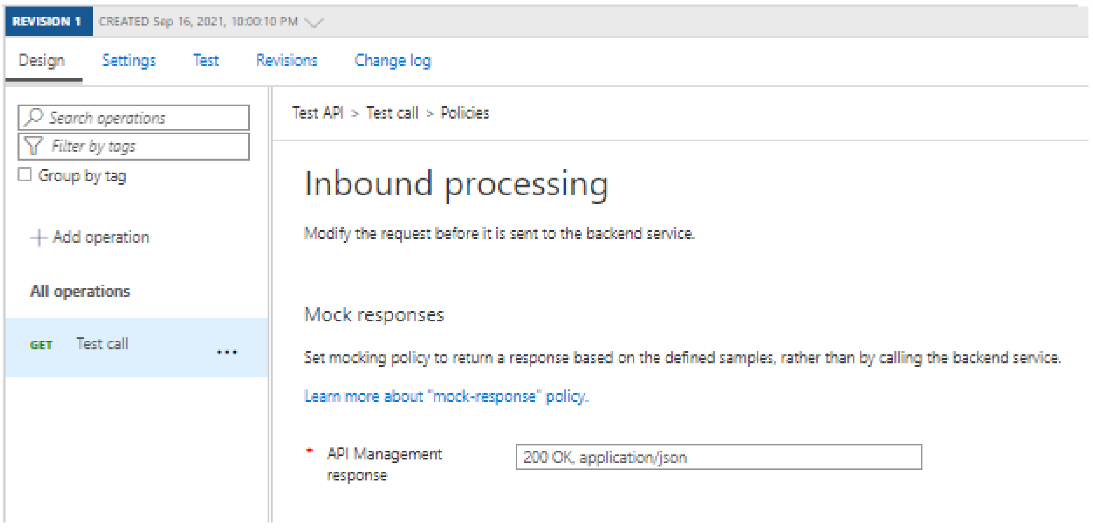
Description

Representations

CONTENT TYPE	SAMPLE	DEFINITION
application/json	{ "sampleField": "test"}	Select defi

Add representation

9. Under Test call, select **+ Add policy** under **Inbound processing**
   Here, we are setting a policy at the specific API scope (as opposed to the product scope we did before)
10. Select **Mock responses**
11. Select **Save**



Group by tag

Add operation

All operations

GET Test call

Test API > Test call > Policies

Inbound processing

Modify the request before it is sent to the backend service.

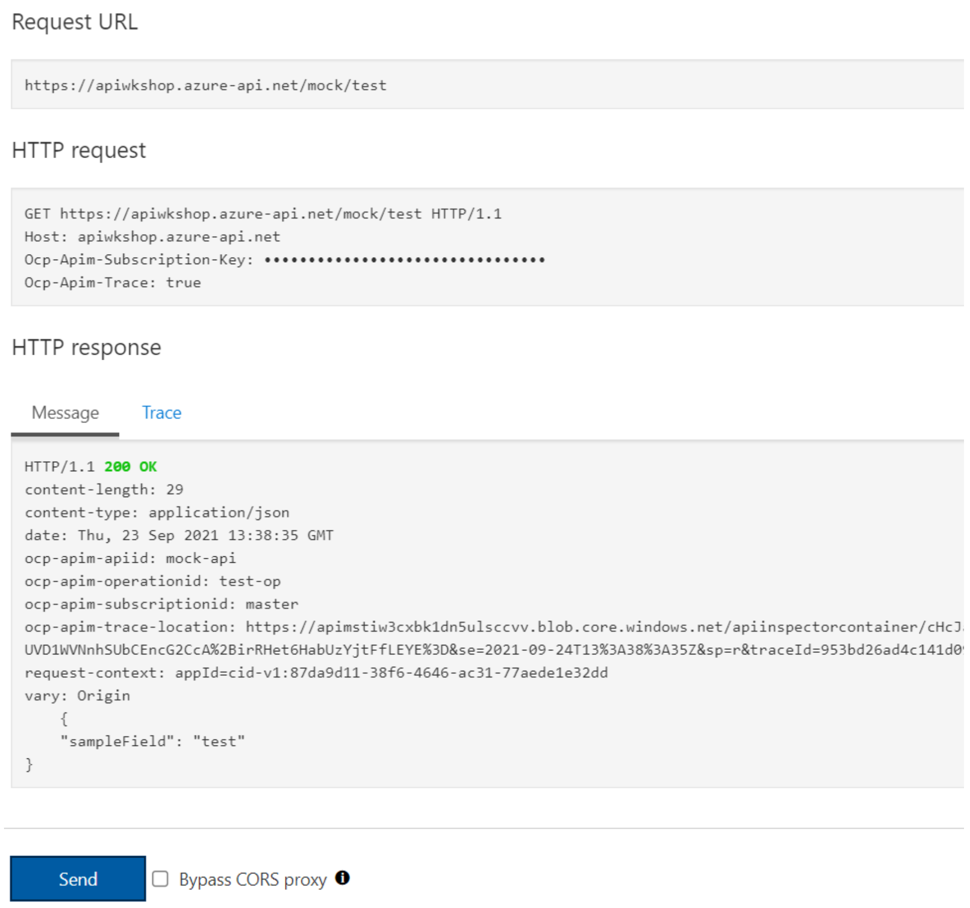
Mock responses

Set mocking policy to return a response based on the defined samples, rather than by calling the backend service.

Learn more about "mock-response" policy.

API Management response: 200 OK, application/json

12. Select the **Test** tab from the top of the screen.
13. Select **Send**



Request URL

```
https://apiwkshop.azure-api.net/mock/test
```

HTTP request

```
GET https://apiwkshop.azure-api.net/mock/test HTTP/1.1
Host: apiwkshop.azure-api.net
Ocp-Apim-Subscription-Key: ••••••••••••••••••••••••••••••••
Ocp-Apim-Trace: true
```

HTTP response

```
HTTP/1.1 200 OK
content-length: 29
content-type: application/json
date: Thu, 23 Sep 2021 13:38:35 GMT
ocp-apim-apiid: mock-api
ocp-apim-operationid: test-op
ocp-apim-subscriptionid: master
ocp-apim-trace-location: https://apimstiw3cxbk1dn5ulsccvv.blob.core.windows.net/apiinspectorcontainer/cHcJ
UVD1WVNnhSUbCEncG2CcA%2BirRHet6HabUzYjtFfLEYE%3D&se=2021-09-24T13%3A38%3A35Z&sp=r&traceId=953bd26ad4c141d0
request-context: appId=cid-v1:87da9d11-38f6-4646-ac31-77aede1e32dd
vary: Origin
    {
    "sampleField": "test"
}
```

Note the 200 response and the data matches the sample we created.

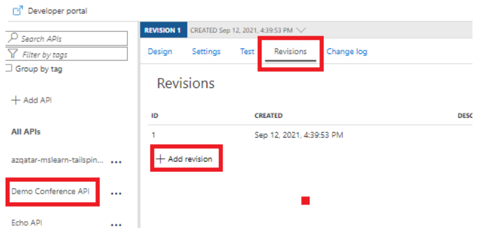
Lab 05: Versions and Revisions

Revisions are for **non-breaking** changes to your API. You can test your features without publishing them. Once you're ready to publish, you can add it to the changelog.

Versions for **breaking** changes to your API. You can run multiple versions simultaneously, with a distinguishing URL path, header, or query string.

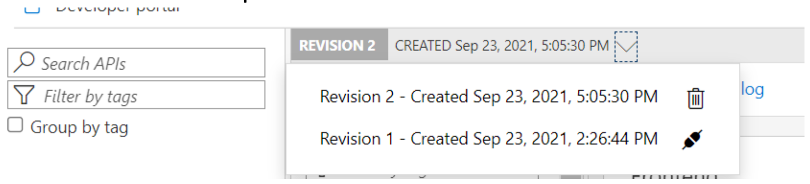
Create a new revision

1. Navigate to the **Demo Conference API**
2. Select the **Revisions** tab
3. Click on **+ Add Revision**
4. Add a description to the revision (for example, Revision 2) and click **Create**
   This is now your revision #2. This is online, but not the CURRENT version. There's also a difference in the URL. (/;rev=2)



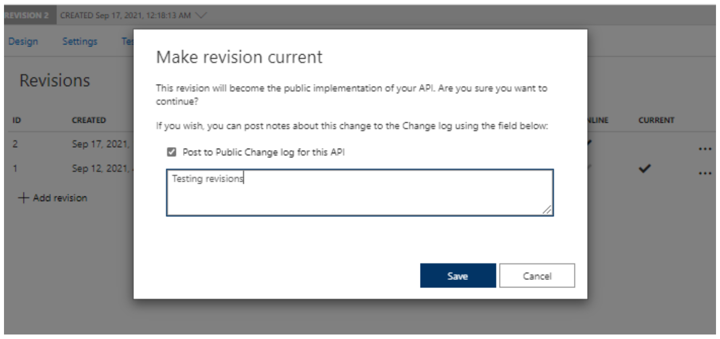
Make a non-breaking change to the revision

1. Select **Demo Conference API**
2. Select the **Design** tab
3. Click **+ Add operation**
4. Enter the following details:
   a. Display name: Test
   b. URL: Get /test
5. Click on **Save**
6. You'll see this operation appear under Revision 2 only. You can validate this by checking the Revision selector drop-down.


Search APIs
Filter by tags
Group by tag
REVISION 2
CREATED Sep 23, 2021, 5:05:30 PM
Revision 2 - Created Sep 23, 2021, 5:05:30 PM
Revision 1 - Created Sep 23, 2021, 2:26:44 PM
log

Make the revision current

1. Select the **Revisions** tab
2. Select the **"…"** ellipsis for **Revision 2**
3. Click **Make Current**
4. Select the **Post to Public Change log for this API** checkbox.



Make revision current

This revision will become the public implementation of your API. Are you sure you want to continue?

If you wish, you can post notes about this change to the Change log using the field below:

☑ Post to Public Change log for this API

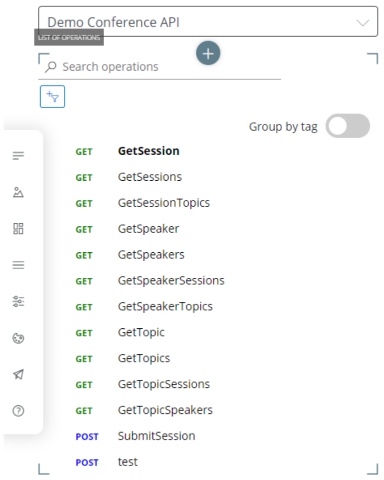
Testing revisions

ID	CREATED	ONLINE	CURRENT
2	Sep 17, 2021,	✓	
1	Sep 12, 2021,	✓	✓

View the change log in developer portal

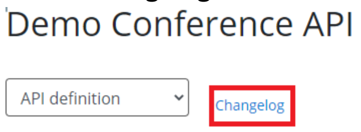
1. Navigate to the **Developer Portal** in a new tab
2. Select **APIs > Demo Conference API**
3. Notice that **test** is now available

contoso



GET GetSession
GET GetSessions
GET GetSessionTopics
GET GetSpeaker
GET GetSpeakers
GET GetSpeakerSessions
GET GetSpeakerTopics
GET GetTopic
GET GetTopics
GET GetTopicSessions
GET GetTopicSpeakers
POST SubmitSession
POST test

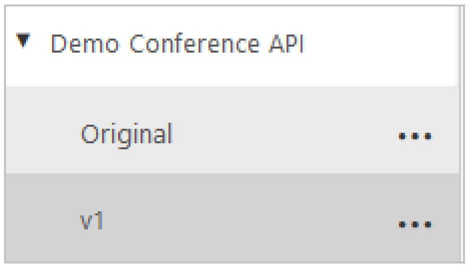
4. Select **Changelog** near the API name for viewing the log entry



Add a new version to the API

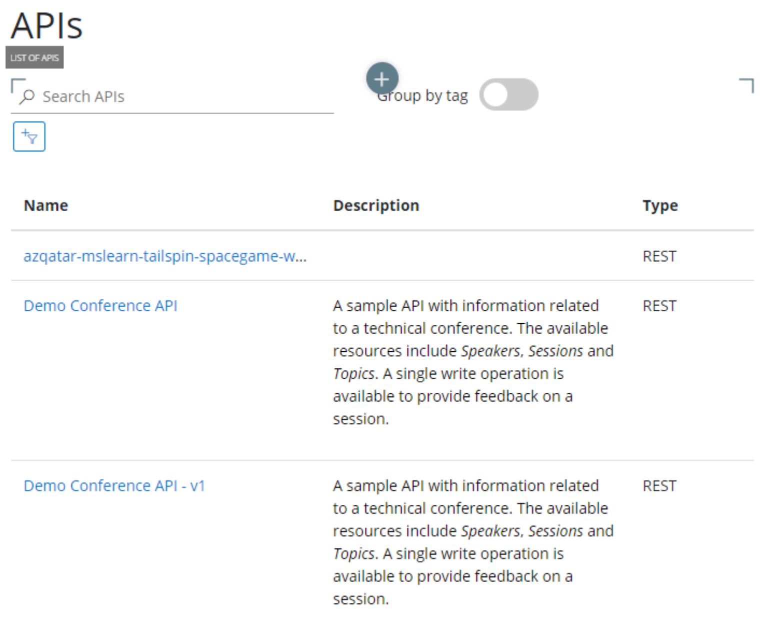
1. Navigate back to **APIs** under the APIM instance.
2. Select the **Demo Conference API**
3. Select the "…" ellipsis
4. Select **Add Version**
    a. Versioning identifier: **v1**
    b. Versioning Scheme: **Path**
    c. API Version name: **demo-conference-api-v1**
    d. Products: **Unlimited, Basic, Starter,** and the product you created earlier.

5. Click **Create**
6. Note that the Demo Conference API now has a dropdown showing **Original** and **v1**



See the new version in the Developer Portal

1. Navigate back to the browser tab with the Developer Portal.
2. Select **APIs.**
3. Note that Demo Conference API displays both **Original** and **v1.**

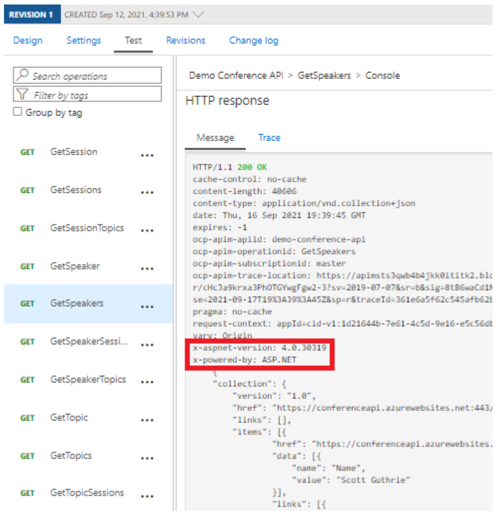


Name	Description	Type
azqatar-mslearn-tailspin-spacegame-w...		REST
Demo Conference API	A sample API with information related to a technical conference. The available resources include *Speakers*, *Sessions* and *Topics*. A single write operation is available to provide feedback on a session.	REST
Demo Conference API - v1	A sample API with information related to a technical conference. The available resources include *Speakers*, *Sessions* and *Topics*. A single write operation is available to provide feedback on a session.	REST

Lab 06: Add Policies

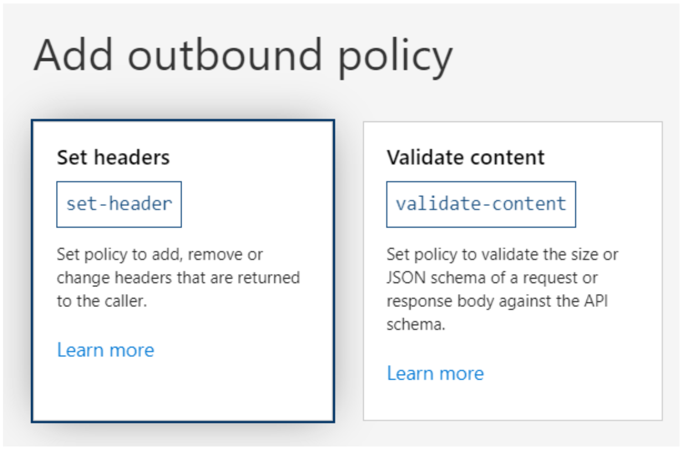
Add policy to strip HTTP headers.

1. Select the **Demo Conference API**
2. Select the **Test** tab.
3. (In the original version), Select the **GetSpeakers** API call
4. Click **Send**
5. Under message, note the “x-aspnet-version” and “x-powered-by” headers. Let us remove these headers from the response (for all operations).



```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 40606
content-type: application/vnd.collection+json
date: Thu, 16 Sep 2021 19:39:45 GMT
expires: -1
ocp-apim-apiid: demo-conference-api
ocp-apim-operationid: GetSpeakers
ocp-apim-subscriptionid: master
ocp-apim-trace-location: https://apimsts3qwb4b4jkk0ititk2.blo
r/cHcJa9krxa3PhOTGYwgFgw2-3?sv=2019-07-07&sr=b&sig=8tB6waCd1N
se=2021-09-17T19%3A39%3A45Z&sp=r&traceId=361e6a5f62c545afb62b
pragma: no-cache
request-context: appId=cid-v1:1d21644b-7e61-4c5d-9e16-e5c56db
vary: Origin
x-aspnet-version: 4.0.30319
x-powered-by: ASP.NET
    {
    "collection": {
        "version": "1.0",
        "href": "https://conferenceapi.azurewebsites.net:443/
        "links": [],
        "items": [{
                "href": "https://conferenceapi.azurewebsites.
                "data": [{
                    "name": "Name",
                    "value": "Scott Guthrie"
                }],
                "links": [{
```

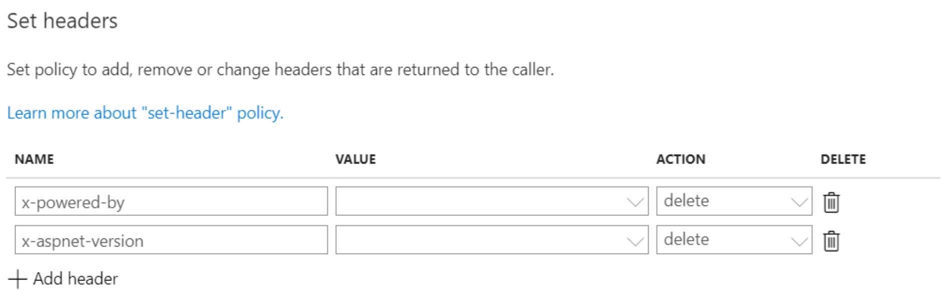
6. Go to the **Design** tab.
7. Select **All operations.**
8. Under **Outbound processing,** click on (+ Add policy)
9. Click on **Set headers** under Transformation policies


Add outbound policy

Set headers
`set-header`
Set policy to add, remove or change headers that are returned to the caller.
Learn more

Validate content
`validate-content`
Set policy to validate the size or JSON schema of a request or response body against the API schema.
Learn more

10. Add the following values:
    a. Name: x-powered-by Action: delete
    b. Name: x-aspnet-version Action: delete



Set headers

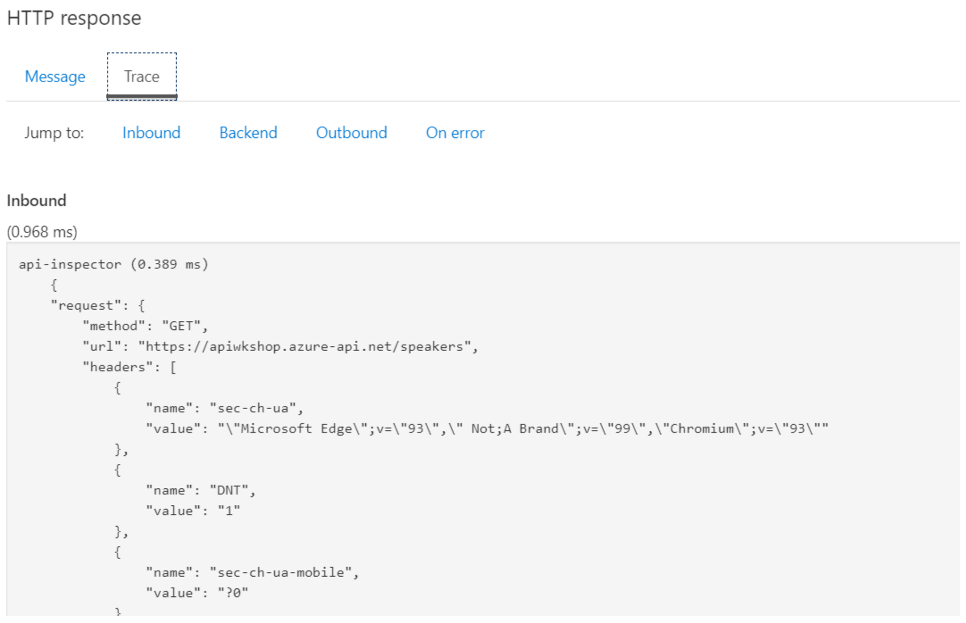
Set policy to add, remove or change headers that are returned to the caller.

Learn more about "set-header" policy.

NAME	VALUE	ACTION	DELETE
x-powered-by		delete	
x-aspnet-version		delete	

+ Add header

11. Select **Save**
12. Under the **Test** tab, test GetSpeakers again. Note that the HTTP response headers are not there anymore.
13. In the response, click on the "Trace" tab to view the various actions performed by API processing pipeline. Browse through the Inbound, Backend, Outbound, On Error sections.



HTTP response

Message   Trace

Jump to: Inbound   Backend   Outbound   On error

Inbound

(0.968 ms)

```
api-inspector (0.389 ms)
    {
    "request": {
        "method": "GET",
        "url": "https://apiwkshop.azure-api.net/speakers",
        "headers": [
            {
                "name": "sec-ch-ua",
                "value": "\"Microsoft Edge\";v=\"93\",\" Not;A Brand\";v=\"99\",\"Chromium\";v=\"93\""
            },
            {
                "name": "DNT",
                "value": "1"
            },
            {
                "name": "sec-ch-ua-mobile",
                "value": "?0"
```

Note that there are more policies that can be applied through the code editor.

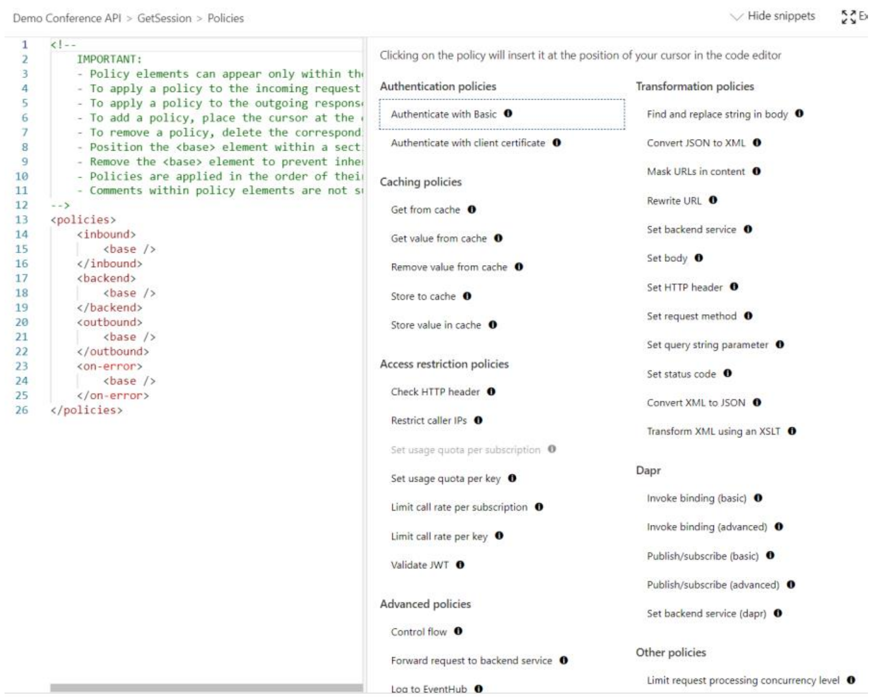
Add policy to mask destination URL in content

Note that in the response, the destination URL conferencewebapi.azurewebsites.net is visible. For security reasons it is desirable to hide the destination URL. Let us add a policy to do that.

14. Click on All Operations. Click on the </> icon next to the **Outbound** processing rules to open the policy code editor.

Policies </>

15. Position the cursor inside the **<outbound>** element and select **Show snippets** at the top right corner.
16. In the right window, under **Transformation policies**, select **Mask URLs in content**.


Demo Conference API > GetSession > Policies

Hide snippets

```
<!--
    IMPORTANT:
    - Policy elements can appear only within th
    - To apply a policy to the incoming request
    - To apply a policy to the outgoing respons
    - To add a policy, place the cursor at the 
    - To remove a policy, delete the correspond
    - Position the <base> element within a sect
    - Remove the <base> element to prevent inhe
    - Policies are applied in the order of thei
    - Comments within policy elements are not s
-->
<policies>
    <inbound>
        <base />
    </inbound>
    <backend>
        <base />
    </backend>
    <outbound>
        <base />
    </outbound>
    <on-error>
        <base />
    </on-error>
</policies>
```

Clicking on the policy will insert it at the position of your cursor in the code editor

Authentication policies
- Authenticate with Basic
- Authenticate with client certificate

Caching policies
- Get from cache
- Get value from cache
- Remove value from cache
- Store to cache
- Store value in cache

Access restriction policies
- Check HTTP header
- Restrict caller IPs
- Set usage quota per subscription
- Set usage quota per key
- Limit call rate per subscription
- Limit call rate per key
- Validate JWT

Advanced policies
- Control flow
- Forward request to backend service
- Log to EventHub

Transformation policies
- Find and replace string in body
- Convert JSON to XML
- Mask URLs in content
- Rewrite URL
- Set backend service
- Set body
- Set HTTP header
- Set request method
- Set query string parameter
- Set status code
- Convert XML to JSON
- Transform XML using an XSLT

Dapr
- Invoke binding (basic)
- Invoke binding (advanced)
- Publish/subscribe (basic)
- Publish/subscribe (advanced)
- Set backend service (dapr)

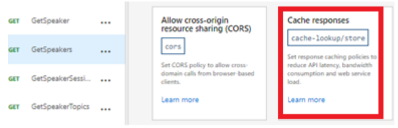
Other policies
- Limit request processing concurrency level

```
<outbound>
    <base />
    <set-header name="x-powered-by" exists-action="delete" />
    <set-header name="x-aspnet-version" exists-action="delete" />
    <redirect-content-urls />
</outbound>
```

17. Click “Save” to save the changes.
18. Test the Get Speakers API again. You will see that all the destination URLs are replaced by Azure API management URLs.

Lab 07: API Caching

1. For **Demo Conference API**, select the **Design** tab.
2. Select the **GetSpeakers API**
3. Under **Inbound processing**, select **+ Add Policy**
4. Select **Cache responses** (cache-lookup/store)



Allow cross-origin resource sharing (CORS)

cors

Set CORS policy to allow cross-domain calls from browser-based clients.

Learn more

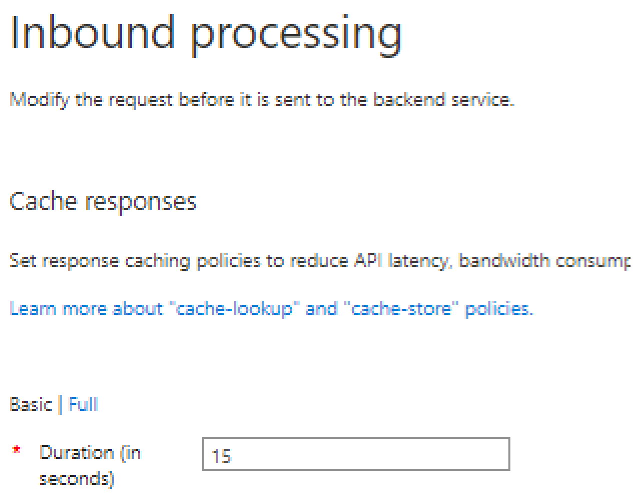
Cache responses

cache-lookup/store

Set response caching policies to reduce API latency, bandwidth consumption and web service load.

Learn more

5. Set the duration to **150 seconds**



Inbound processing

Modify the request before it is sent to the backend service.

Cache responses

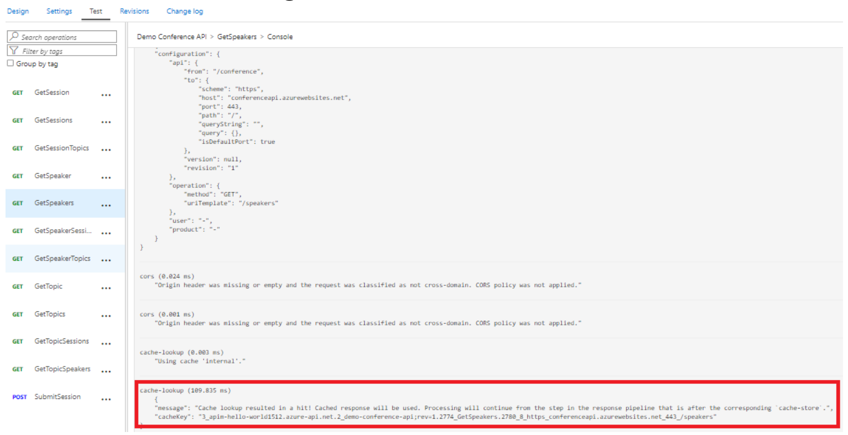
Set response caching policies to reduce API latency, bandwidth consump

Learn more about "cache-lookup" and "cache-store" policies.

Basic | Full

* Duration (in seconds) 15

6. Select **Save**
7. Select the **Test** tab for Demo Conference API
8. Select **GetSpeakers** and send a test to the API
9. Go to the **Trace** tab of the response. You'll notice that the cache was a miss.
10. Send a test to the API again and check the **Trace** tab. You'll see a hit!



```
    "configuration": {
        "api": {
            "from": "/conference",
            "to": {
                "scheme": "https",
                "host": "conferenceapi.azurewebsites.net",
                "port": 443,
                "path": "/",
                "queryString": "",
                "query": {},
                "isDefaultPort": true
            },
            "version": null,
            "revision": "1"
        },
        "operation": {
            "method": "GET",
            "uriTemplate": "/speakers"
        },
        "user": "-",
        "product": "-"
    }
}

cors (0.024 ms)
    "Origin header was missing or empty and the request was classified as not cross-domain. CORS policy was not applied."

cors (0.001 ms)
    "Origin header was missing or empty and the request was classified as not cross-domain. CORS policy was not applied."

cache-lookup (0.003 ms)
    "Using cache 'internal'."

cache-lookup (109.835 ms)
    {
    "message": "Cache lookup resulted in a hit! Cached response will be used. Processing will continue from the step in the response pipeline that is after the corresponding `cache-store`.",
    "cacheKey": "3_apim-hello-world1512.azure-api.net.2_demo-conference-api;rev=1.2774_GetSpeakers.2780_8_https_conferenceapi.azurewebsites.net_443_/speakers"
```

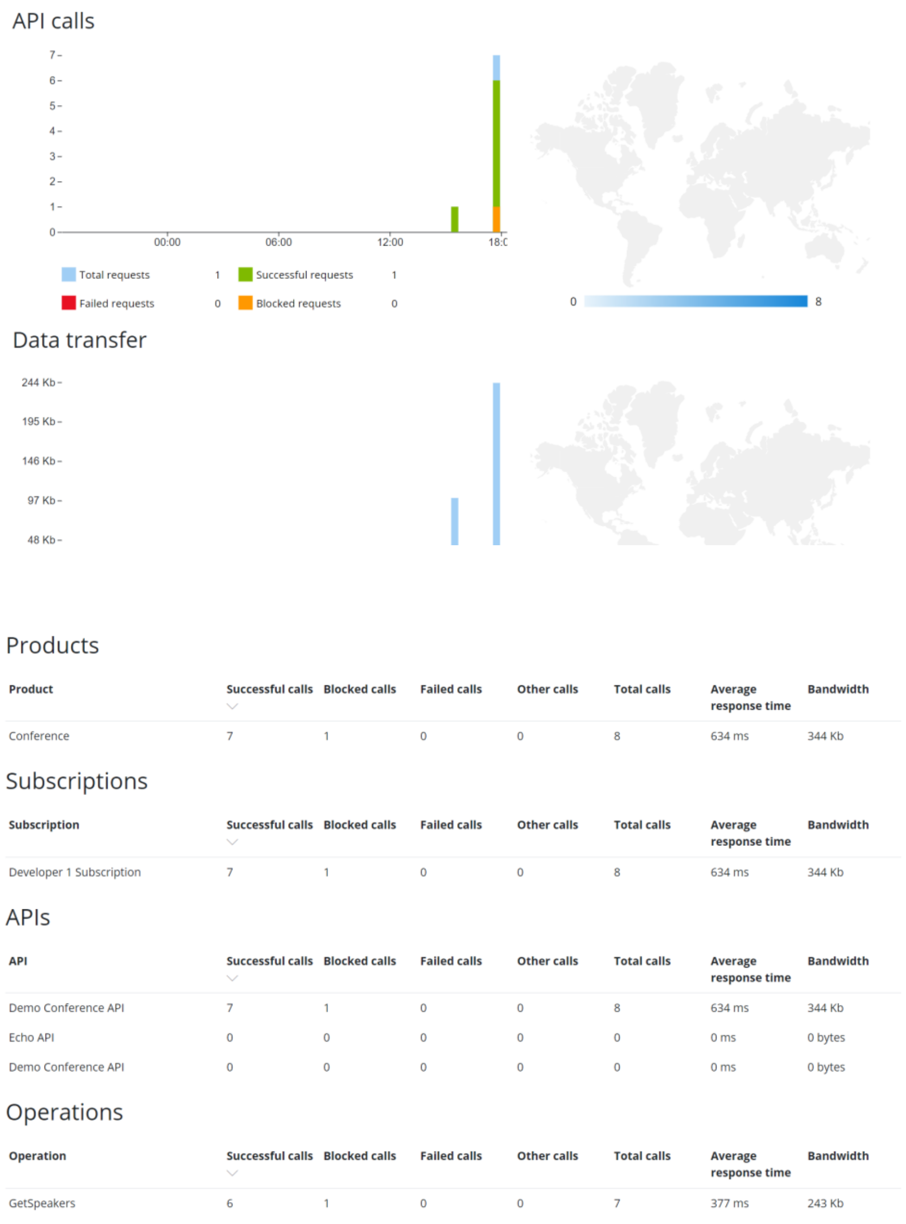
An important resource here is the ability to use an external Redis-compatible cache. Read more about how to implement it here: Use an external cache in Azure API Management | Microsoft Docs

Lab 08: API Monitoring

API dashboards in developer portal.

As an API developer, you will be able to view the API statistics to which you have access to.

1. In the browser window where developer portal is open, click on the “Reports” menu.
2. Select the duration as Today (or Last Hour).
3. Scroll down and view the API invocation statistics / reports.



Products

Product	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Average response time	Bandwidth
Conference	7	1	0	0	8	634 ms	344 Kb

Subscriptions

Subscription	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Average response time	Bandwidth
Developer 1 Subscription	7	1	0	0	8	634 ms	344 Kb

APIs

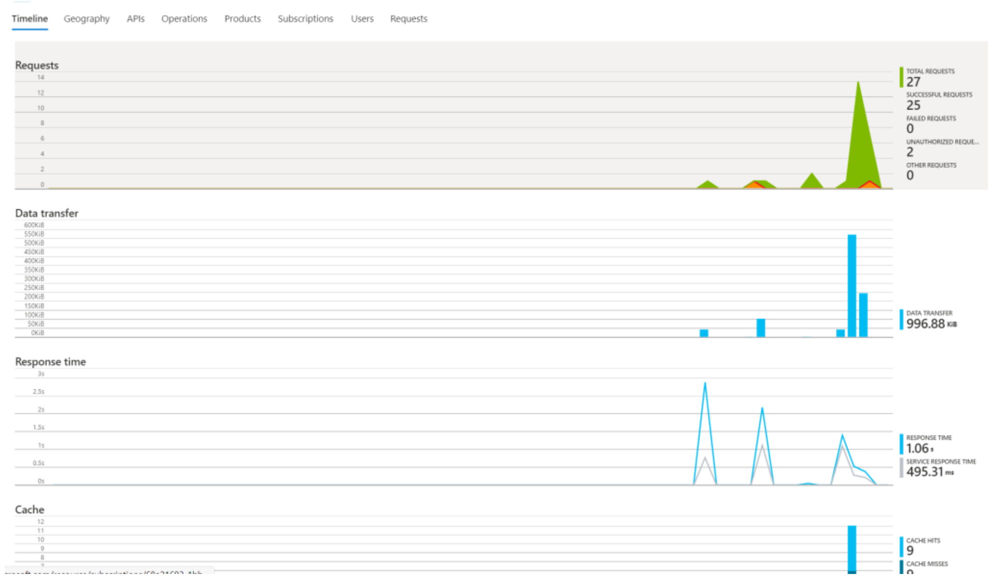
API	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Average response time	Bandwidth
Demo Conference API	7	1	0	0	8	634 ms	344 Kb
Echo API	0	0	0	0	0	0 ms	0 bytes
Demo Conference API	0	0	0	0	0	0 ms	0 bytes

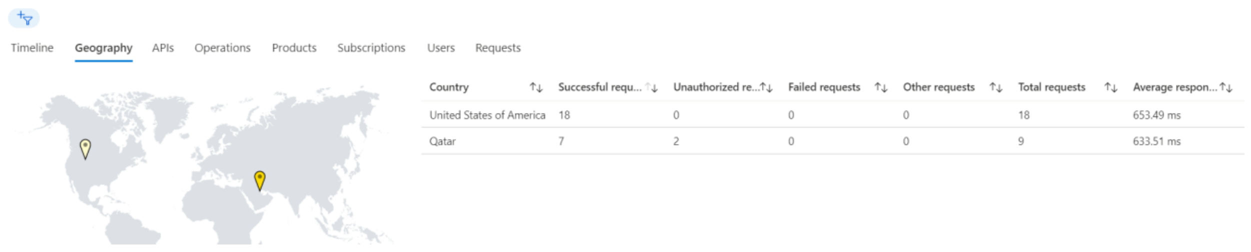
Operations

Operation	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Average response time	Bandwidth
GetSpeakers	6	1	0	0	7	377 ms	243 Kb
]

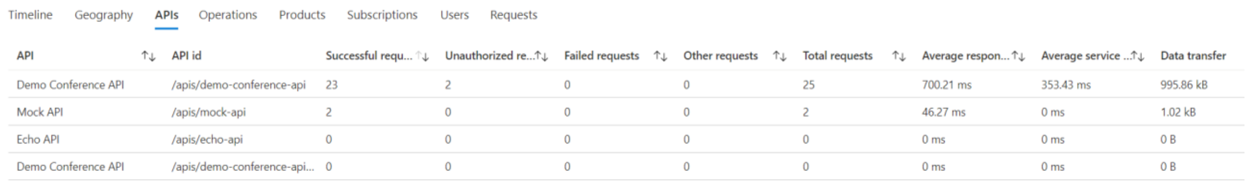
API dashboards in Azure API management

4. Go back to the browser with Azure API management open, and select Monitoring -> Analytics
5. Observe the values in various tabs.





Country	Successful requ...	Unauthorized re...	Failed requests	Other requests	Total requests	Average respon...
United States of America	18	0	0	0	18	653.49 ms
Qatar	7	2	0	0	9	633.51 ms

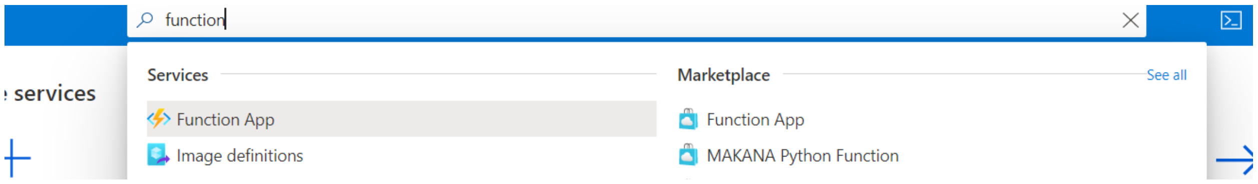


API	API id	Successful requ...	Unauthorized re...	Failed requests	Other requests	Total requests	Average respon...	Average service ...	Data transfer
Demo Conference API	/apis/demo-conference-api	23	2	0	0	25	700.21 ms	353.43 ms	995.86 kB
Mock API	/apis/mock-api	2	0	0	0	2	46.27 ms	0 ms	1.02 kB
Echo API	/apis/echo-api	0	0	0	0	0	0 ms	0 ms	0 B
Demo Conference API	/apis/demo-conference-api...	0	0	0	0	0	0 ms	0 ms	0 B

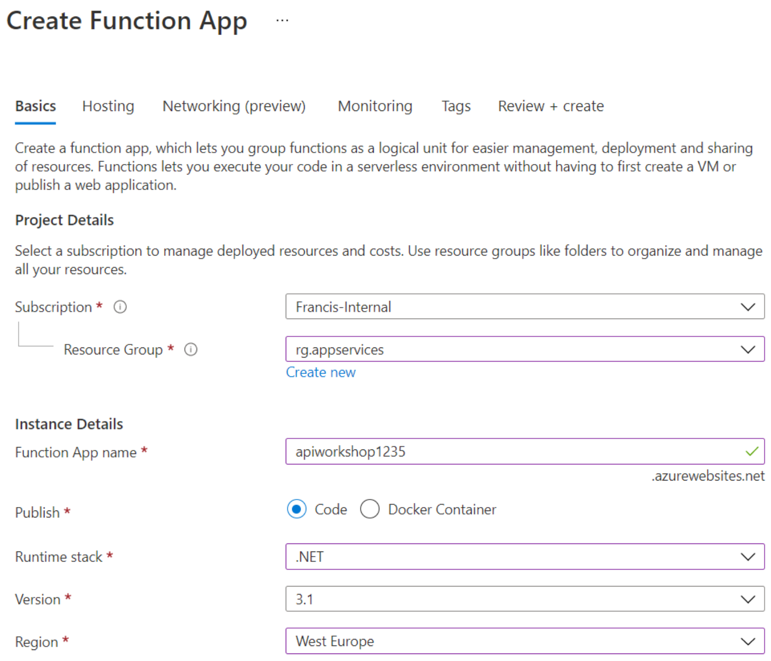
Lab 09: Exposing an Azure Function as an API

Create a Function App.

1. In Azure Portal, search for Function and select “Function App”.



2. Click (+ Create) to create a function app.
3. Select a resource group, region, a unique name for the function app, and runtime stack (.NET) with version 3.1



Create Function App

Create a function app, which lets you group functions as a logical unit for easier management, deployment and sharing of resources. Functions lets you execute your code in a serverless environment without having to first create a VM or publish a web application.

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

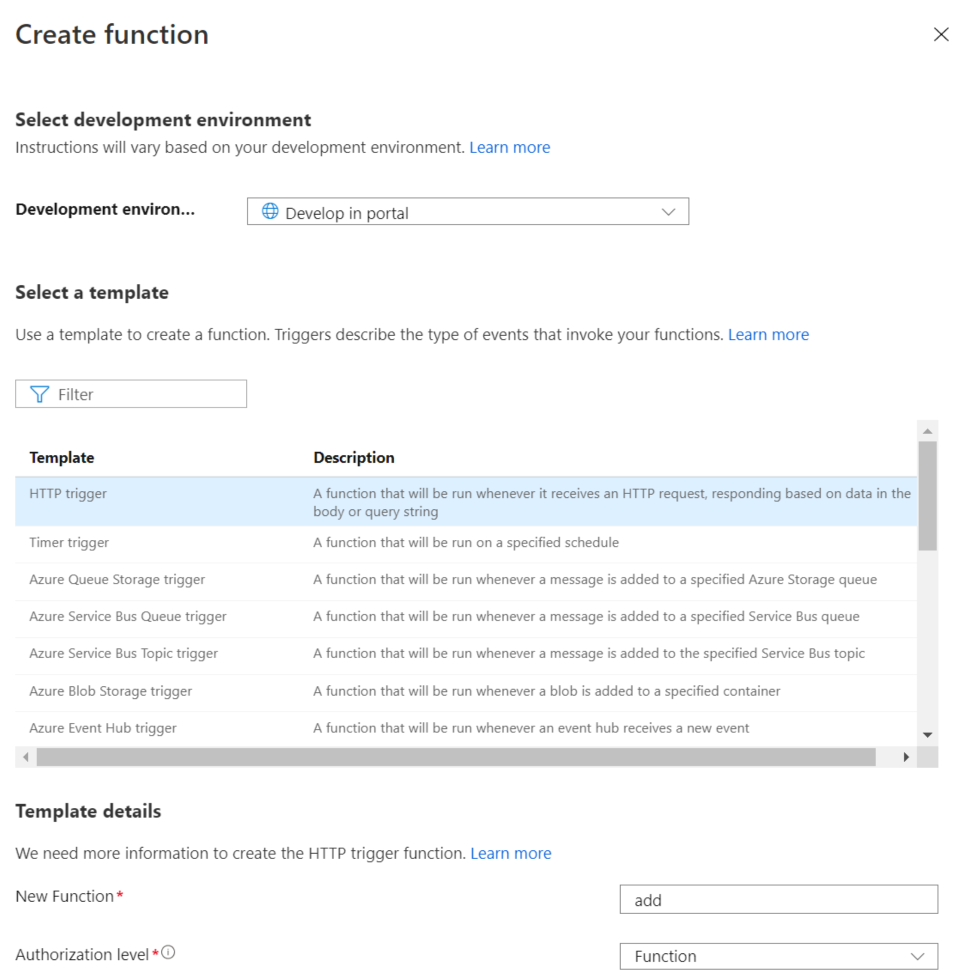
Subscription *	Francis-Internal
Resource Group *	rg.appservices

Create new

Instance Details

Function App name *	apiworkshop1235 .azurewebsites.net
Publish *	[✓ Code] Docker Container
Runtime stack *	.NET
Version *	3.1
Region *	West Europe

4. Click "Review & Create", and then "Create" the function app.
5. Once the deployment is complete, click on "Go to resource".
6. From the left ribbon, click "Functions".
7. Click on +Create to create a function.
8. Select the template as "HTTP Trigger", and rename the new function name to "add".


Create function

Select development environment

Instructions will vary based on your development environment. Learn more

Development environ...	Develop in portal

Select a template

Use a template to create a function. Triggers describe the type of events that invoke your functions. Learn more

Template	Description
HTTP trigger	A function that will be run whenever it receives an HTTP request, responding based on data in the body or query string
Timer trigger	A function that will be run on a specified schedule
Azure Queue Storage trigger	A function that will be run whenever a message is added to a specified Azure Storage queue
Azure Service Bus Queue trigger	A function that will be run whenever a message is added to a specified Service Bus queue
Azure Service Bus Topic trigger	A function that will be run whenever a message is added to the specified Service Bus topic
Azure Blob Storage trigger	A function that will be run whenever a blob is added to a specified container
Azure Event Hub trigger	A function that will be run whenever an event hub receives a new event

Template details

We need more information to create the HTTP trigger function. Learn more

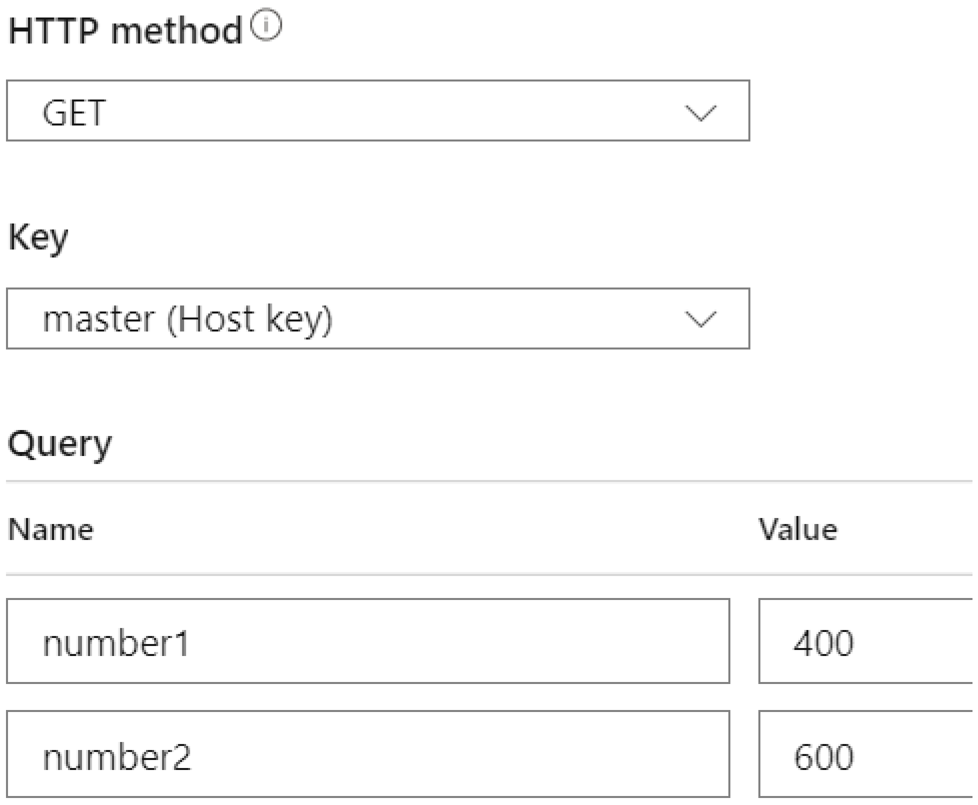
New Function*	add
Authorization level*	Function

9. Click “Create” to create a function within the function app.
10. In the function view, click on “Code + Test” under the Developer menu.
11. Replace the code in the block with the following code:

```
#r "Newtonsoft.Json"

using System.Net;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Primitives;
using Newtonsoft.Json;

public static async Task<IActionResult> Run(HttpRequest req, ILogger log)
{
    string number1 = req.Query["number1"];
    string number2 = req.Query["number2"];

    string responseMessage = string.IsNullOrEmpty(number1) ||
string.IsNullOrEmpty(number2)
```

```
        ? "This HTTP triggered function executed successfully. Pass number1 and 
number2 in the query string."
                : "" + ( int.Parse(number1) + int.Parse(number2) );

            return new OkObjectResult(responseMessage);
}
```

12. Click on the “Test / Run” button.
13. Change the HTTP method to “Get”
14. Add two Query parameters: number1 and number2 with two integer values.


HTTP method: GET

Key: master (Host key)

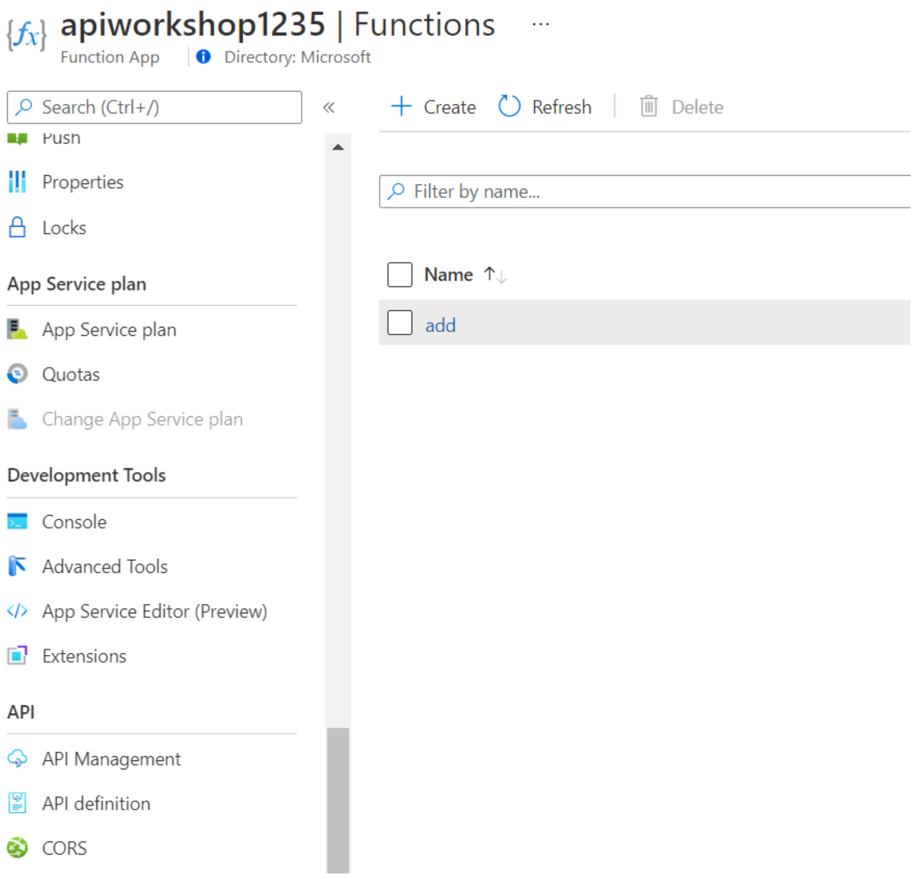
Query

Name	Value
number1	400
number2	600

15. Click “Run”. You should see the result of addition.

Expose the function as an API.

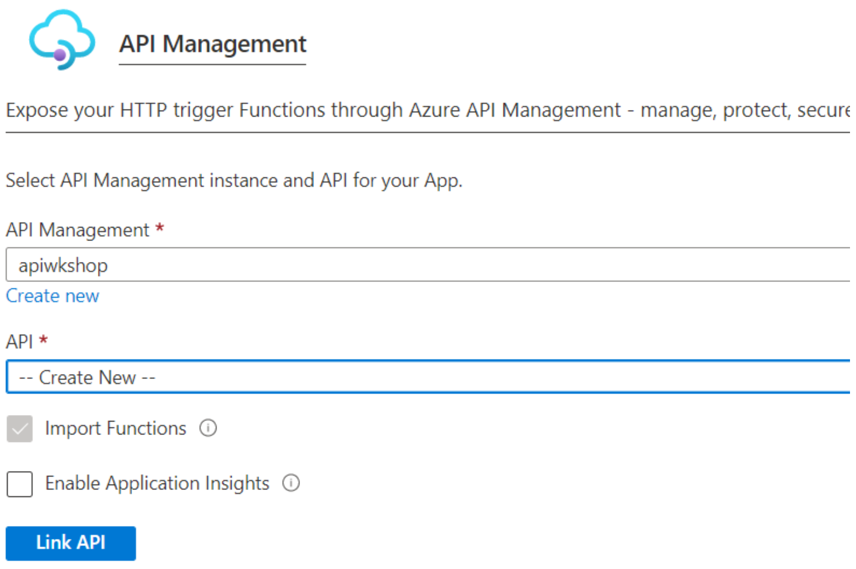
16. Navigate back to the Functions view.
17. On the left side ribbon, scroll down to the “API” section.



apiworkshop1235 | Functions

Function App | Directory: Microsoft

18. Click on “API Management”.
19. Select the API management instance you created, and for the API, leave the “Create New” option.



API Management

Expose your HTTP trigger Functions through Azure API Management - manage, protect, secure

Select API Management instance and API for your App.

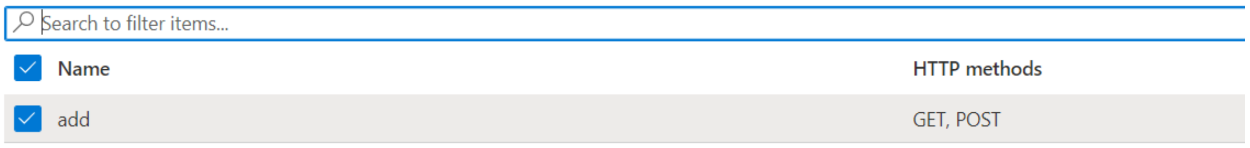
API Management *
apiwkshop
Create new

API *
-- Create New --

[✓] Import Functions

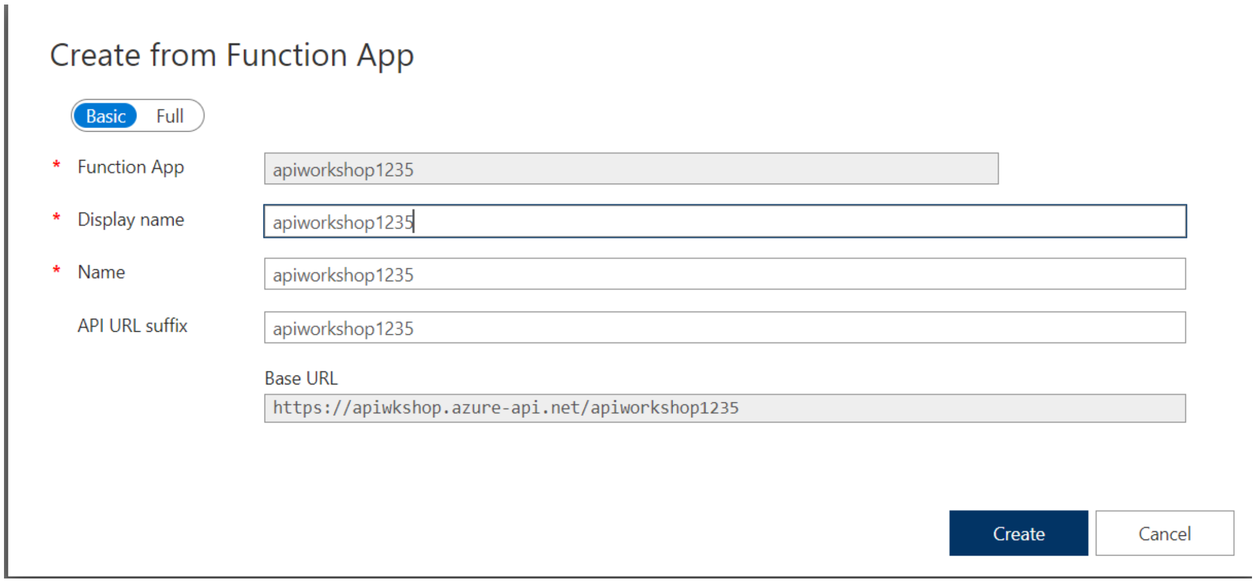
[ ] Enable Application Insights

20. Click on the “Link API” button.
21. Select the “add” function and click “Select”



Name	HTTP methods
✓ add	GET, POST

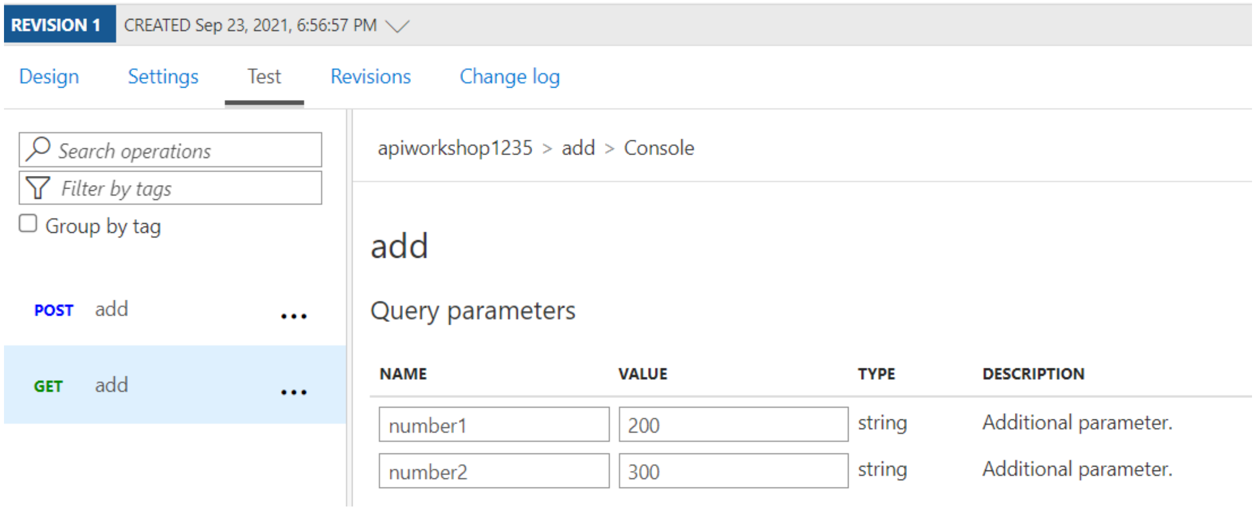
22. Select the defaults (or change values) in the “Create from function app” dialog box.



Create from Function App

* Function App	apiworkshop1235
* Display name	apiworkshop1235
* Name	apiworkshop1235
API URL suffix	apiworkshop1235
Base URL	https://apiwkshop.azure-api.net/apiworkshop1235

23. Click “Create” to create the API.
24. Once the API is created, click on “Go to API management”.
25. From the APIs, select the API that got created (for example, apiworkshop1235) and view the Get operation.
26. Inspect the inbound policies, and the backend policy.
27. Click on the “Test” tab to test the API.
28. On the query parameters, add two parameters (number1 and number2) with two integer values.



REVISION 1 CREATED Sep 23, 2021, 6:56:57 PM

apiworkshop1235 > add > Console

add

Query parameters

NAME	VALUE	TYPE	DESCRIPTION
number1	200	string	Additional parameter.
number2	300	string	Additional parameter.

29. Click “Send” to test the API.
30. You should see the add function’s results.

Optional Labs

Optional Lab 01: APIM with Azure DevOps

azure-apim-lab/apimanagement-8.md at master · feranto/azure-apim-lab (github.com)

Optional Lab 02: Integrate APIM with Virtual Networks

Connect to a virtual network using Azure API Management | Microsoft Docs

Alternatively, for an internal virtual network:

Connect to an internal virtual network using Azure API Management | Microsoft Docs

Optional Lab 03: Manage APIs in Visual Studio Code

Tutorial - Import and manage APIs - Azure API Management and Visual Studio Code | Microsoft Docs