

Practical Exercise: Creating and Publishing an API through the Publisher

Training Objective

Learn how to create an API, add API documentation, and publish it to the Developer Portal using the Publisher.

Business Scenario

After setting up the API-M, the API is created and published through the API Publisher in order to make it subscribable from the store.

PizzaShack Limited is providing a store from which consumers can subscribe to their API. This works as a secondary business function for PizzaShack and attracts many developers to the PizzaShack website. The API will be comprehensively documented for ease of use.

High Level Steps

- Add the PizzaShack API to the Publisher
- Add documentation
- Publish the APIs

Detailed Instructions

Adding the PizzaShack API to the Publisher

Now that we set up the API-M and added users, we are ready to publish the API the PizzaShack application requires.

To add the API to the publisher, follow those steps:

1. Open the API Publisher web application from <https://localhost:9443/publisher>.
2. Log in using the user with creator role you defined previously (creator).
3. Click **Create API**.
4. Select **Design a New Rest API**.
5. Provide information on the API as per the table below and click **Create**.

Field	Value	Description
Name	PizzaShack	Name of API as you want it to appear in the API store
Context	/pizzashack	URI context path that is used by API consumers (Application Developers)
Version	1.0.0	API version (in the form of version.major.minor)
Endpoint	https://localhost:9443/am/sample/pizzashack/v1/api/	The endpoint that you add is automatically added as the production and sandbox endpoints.
Business Plans	Unlimited, Gold	Attach business plans to API

6. Select **Design Configurations** tab and provide the following information and click **Save**.

Field	Value	Description
Publisher Access Control	All	The ability to protect an API to be managed only by users with specific roles
Developer Portal Visibility	Public	Whether this API is visible to all or restricted to certain roles.
Thumbnail Image	Download a PizzaShack logo image and upload it Get the Logo here : Link	Icon to be displayed in the API store (can be jpeg, tiff, png format).
Description	PizzaShackAPI: Allows to manage pizza orders (create, update, retrieve orders)	High level description of API functionality
Tags	pizza, order, pizza-menu	One of more tags. Tags are used to group/search for APIs (Press Enter after each tag)

Make this default version	No	The default version option allows you to mark one API, from a group of API versions, as the default one, so that it can be invoked without specifying the version number in the URL.
---------------------------	----	--

7. Select **Runtime Configurations** tab and use the following information to configure and click **Save**.

Field	Value	Description
Transports	HTTP/HTTPS	APIs can be exposed in HTTP and/or HTTPS transport: The transport protocol on which the API is exposed. Both HTTP and HTTPS transports are selected by default. If you want to limit API availability to only one transport (e.g., HTTPS), un-check the other transport.
Response Caching	Disabled	Response caching is used to enable caching of response messages per API. Caching protects the backend systems from being exhausted due to serving the same response (for same request) multiple times. If you select the enable option, specify the cache timeout value (in seconds) within which the system tries to retrieve responses from the cache without going to the backend.
Maximum Backend Throughput	Unlimited	Limits the total number of calls the API Manager is allowed to make to the backend. While the other throttling levels define the quota the API invoker gets, they do not ensure that the backend is protected from overuse. Hard throttling limits the quota the backend can handle.
Message Mediation	None	Define your own message mediation policy for incoming and outgoing messages.

CORS Configuration	Not Selected	Enable CORS for the API
--------------------	--------------	-------------------------

8. Select **Resources** tab and define the following resources and click **Save**.

For the PizzaShackAPI, we will be defining 4 resources as defined below.

Resource URI Pattern	HTTP Verb
menu	GET
order	POST
order/{orderid}	GET
order/{orderid}	PUT

The screenshot shows the 'Resources' tab in an API management console. It displays four resource definitions, each with a colored header bar indicating the HTTP verb:

- /menu** (blue header): GET method.
- /order** (green header): POST method.
- /order/{orderid}** (blue header): GET method.
- /order/{orderid}** (orange header): PUT method.

At the bottom of the interface, there are 'SAVE' and 'RESET' buttons, and a link to 'Edit API Definition'.

9. Select **Local Scopes** tab and add a scope with the following information.

For this use case, we will also be making use of OAuth2.0 scopes. Therefore we will be creating a scope named `order_pizza` and allowing that scope only to the users with the admin role.

Scopes enable fine-grained access control to API resources based on user roles. You define scopes to an API's resources. When a user invokes the API, their OAuth 2 bearer token cannot grant access to any API resource beyond its associated scopes.

Field	Description
Scope Name	A unique key for identifying the scope. Typically, it is prefixed by part of the API's name for uniqueness, but is not necessarily reader-friendly.
Roles	The user role(s) that are allowed to obtain a token against this scope. E.g., manager, employee.

To invoke an API protected by scopes, you need to get an access token via the Token API or tokens with specified scopes needs to be generated from the **APPLICATIONS** page in the API Devportal.

Scopes > Create New Scope

Name

order_pizza

Enter Scope Name (E.g., creator)

Description

Only users with admin role and internal/subscriber role can order

Short description about the scope

Roles

admin

internal/subscriber

Enter a valid role and press 'Enter'.

Save

Cancel

Field	Value
Scope Name	order_pizza
Roles	admin, Internal/subscriber

Description	Only users with admin role and internal/subscriber role can order
-------------	---

Since the scope is defined, we need to assign that scope to the appropriate resources

10. Select **Resources** tab again and select **POST /order** resource and add the following information and click **Save**.

Field	Value
Summary	Order Pizza
Description	Create a new Order
Operation Scope	Select "order_pizza"

POST /order Order Pizza Scope : order_pizza

Summary & Description

Description: Create a new Order

Summary: Order Pizza

Operation Governance (Security, Rate Limiting & Scopes)

Security:

Rate limiting policy: Unlimited

Operation scope: order_pizza

Parameters

Parameter Type	Parameter Name	Data Type	Required	Actions
Body	body		Yes	

Add Documentation

1. Select the **Documents** tab and enter the following details to add a document.

Field	Value
Name	PizzaShack
Summary	This is the official documentation for the PizzaShack API
Type	How To
Source	In-line

Documents > Add New Document

Name *

PizzaShack

Provide the name for the document

Summary *

This is the official documentation for the PizzaShack API


Provide a brief description for the document

Type

☒ ? How To ☐ <> Sample & SDK ☐ Public Forum ☐ Support Forum ☐ Other

Source

☒ Inline ☐ Markdown ☐ URL ☐ File

 Please save the document. The content can be edited in the next step.

ADD DOCUMENT CANCEL

Several documents types are available:

- How To
- Samples and SDK
- Public Forum
- Support Forum
- Other

Once the document has been added, you can edit the contents by clicking on the **Edit Content** link. An embedded editor allows you to edit the document contents.

Publish the API

The API is now ready to be published. This has to be done by a user with the publisher role.

To publish the API:

1. Log out as creator and login as publisher user.
2. Click on the API - You can see that an additional tab named **Lifecycle** is now available, allowing you to manage the API lifecycle.
3. To publish the API, select **PUBLISH**.

The API is now published and visible to consumers in the API Devportal. The API life cycle history is visible at the bottom of the page

The screenshot shows the WSO2 API Manager interface for the 'PizzaShackAPI:1.0.0'. The 'Lifecycle' tab is active, displaying a lifecycle diagram with states: CREATED, PROTOTYPED, PUBLISHED, DEPRECATED, RETIRED, and BLOCKED. The 'PUBLISHED' state is highlighted. On the right, a 'Requirements' panel lists conditions for the next state transition. Below this, a 'History' table shows the lifecycle changes.

User	Action	Time
publisher	LC has changed from PUBLISHED to CREATED	a few seconds ago
admin	LC has changed from CREATED to PUBLISHED	2 hours ago

Expected Outcome

The PizzaShackAPI which manages pizza orders has been created and published and can be accessed through the API Devportal.