

Practical Exercise: Working with Rate Limiting Policies

Training Objective

Learn to add new rate limiting policies and define extra properties to rate limiting policies using the Admin Portal. Rate limiting allows you to limit the number of hits to an API during a given period of time.

Business Scenario

PizzaShack's popularity is overwhelming and the amount of requests are increasing so they have decided to allow up to 100 requests per minute. In a recent analysis they have found that PizzaShack API is being misused through an application called "Pizzaman" and they want to block all calls from that application.

High-Level Steps

- Add rate limiting policy
- Add conditions to rate limiting policy
- Block all calls from an application by denying an application.

Detailed Instructions

Add Rate Limiting Policy

1. Log in to the API Manager Admin Portal (<https://localhost:9443/admin/>) (admin/admin).
2. Under **RATE LIMITING POLICIES** select **SUBSCRIPTION POLICIES** and **ADD POLICY** at the top.

WSO2 API ADMIN PORTAL

Rate Limiting Policies / Subscription Policies

Subscription Rate Limiting Policies

Search by Subscription Policy name Add Policy

| Name | Quota Policy | Quota | Unit Time | Rate Limit | Time Unit | Actions |
|-----------------|---------------|------------|-----------|------------|-----------|---|
| Gold | Request Count | 5000 | 1 min | NA | NA | Edit Delete |
| Silver | Request Count | 2000 | 1 min | NA | NA | Edit Delete |
| Bronze | Request Count | 1000 | 1 min | NA | NA | Edit Delete |
| Unauthenticated | Request Count | 500 | 1 min | NA | NA | Edit Delete |
| Unlimited | Request Count | 2147483647 | 1 min | NA | NA | Edit Delete |

Rows per page: 10 1-5 of 5

3. Enter the following information

Subscription Rate Limiting Policy - Create new

General Details

Provide the name and description of the subscription policy.

Name *

Platinum

Name of the rate limiting policy

Description

25 requests per minute

Description of the rate limiting policy

Quota Limits

Request Count and Request Bandwidth are the two options for Quota Limit. You can use the option according to your requirement.



Request Count



Request Bandwidth

Request Count *

25

Number of requests allowed

Unit Time *

1

Unit Time

Minute(s)

Burst Control (Rate Limiting)

Define Burst Control Limits for the subscription policy. This is optional.

Request Rate

5

Number of requests for burst control

Requests/s

GraphQL

Provide the Maximum Complexity and Maximum depth values for GraphQL APIs using this policy.

Max Complexity

Max Depth

Policy Flags

Define the billing plan for the subscription policy. Enable stop on quota reach to block invoking an API when the defined quota is reached.

Billing Plan



Free



Commercial

Stop On Quota Reach



Custom Attributes

Define custom attributes for the subscription policy.



Add Custom Attribute

Permissions

Define the permissions for the subscription policy.

Roles *

Internal/everyone

This policy is "Allowed" for above roles.



Allow



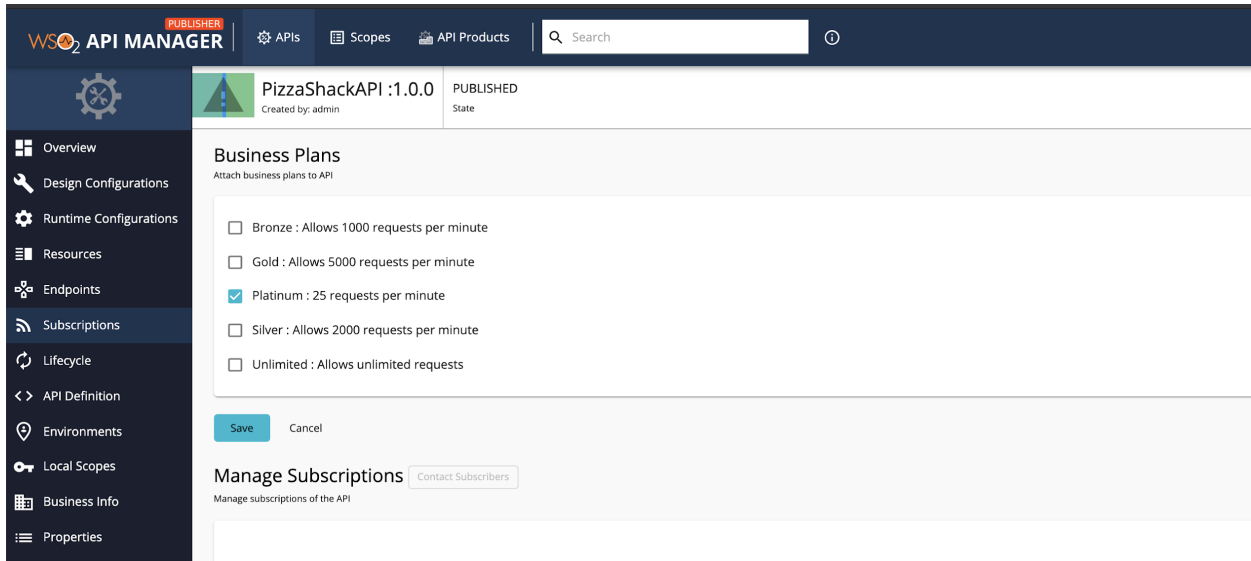
Deny

Save

Cancel

In the API **Publisher**, edit the PizzaShack API and note that Platinum can now be selected, **tick the checkbox of Platinum tier** under **Business Plans** in the **Subscriptions** tab, edit and save it.

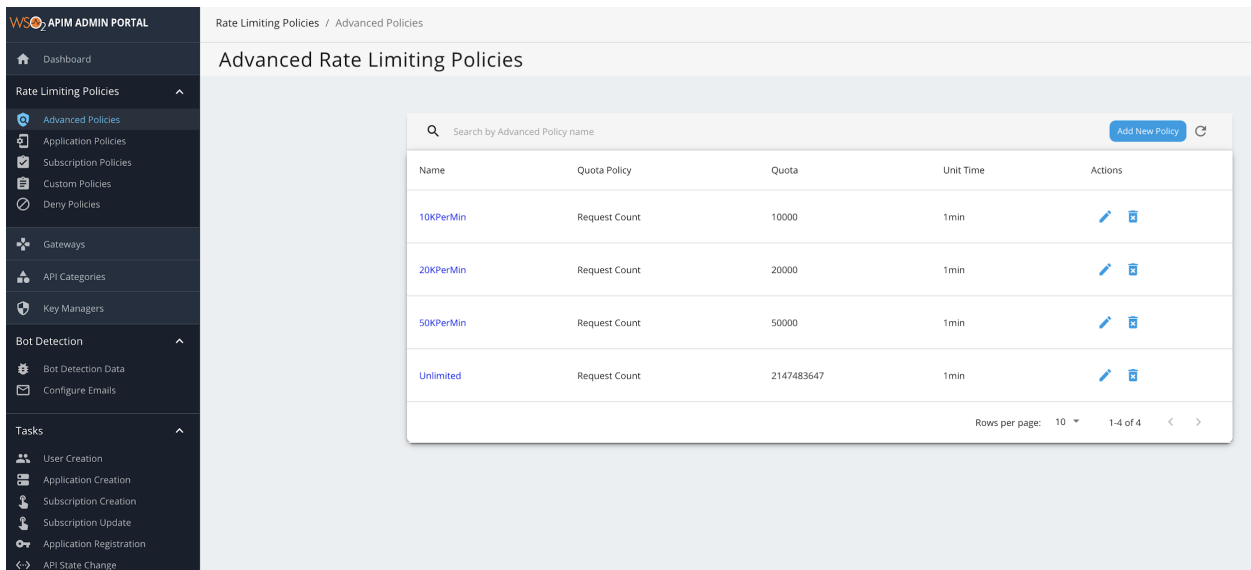
The API will be visible in the API Devportal whenever a person wants to subscribe to the PizzaShack API.



The screenshot shows the WSO2 API Manager Publisher interface. The top navigation bar includes 'APIs', 'Scopes', and 'API Products'. The left sidebar lists various configuration options: Overview, Design Configurations, Runtime Configurations, Resources, Endpoints, Subscriptions, Lifecycle, API Definition, Environments, Local Scopes, Business Info, and Properties. The main content area displays the 'PizzaShackAPI :1.0.0' configuration, which is in a 'PUBLISHED' state. Under the 'Business Plans' section, there are four plans: Bronze (1000 requests per minute), Gold (5000 requests per minute), Platinum (25 requests per minute, selected), and Silver (2000 requests per minute). Below the plans, there are 'Save' and 'Cancel' buttons. The 'Manage Subscriptions' section is also visible, with a 'Contact Subscribers' button.

Add Conditions to Advanced Rate Limiting

1. Click **ADVANCED POLICIES** and select **ADD POLICY** at the top



The screenshot shows the WSO2 API Admin Portal interface. The left sidebar lists various configuration options: Dashboard, Rate Limiting Policies, Application Policies, Subscription Policies, Custom Policies, Deny Policies, Gateways, API Categories, Key Managers, Bot Detection, and Tasks. The main content area displays the 'Advanced Rate Limiting Policies' configuration page. At the top, there is a search bar and an 'Add New Policy' button. Below this, there is a table with the following columns: Name, Quota Policy, Quota, Unit Time, and Actions. The table contains four rows of policies: 10KPerMin, 20KPerMin, 50KPerMin, and Unlimited. Each row has a 'Request Count' quota policy and a '1min' unit time. The 'Unlimited' policy has a quota of 2147483647. The 'Actions' column contains edit and delete icons for each policy. At the bottom of the table, there is a 'Rows per page' dropdown set to 10 and a '1-4 of 4' pagination indicator.

2. Enter the following information and click **Add Conditional Group**. Select **Header Condition** and add the details as shown in the image.

The conditional group is created to apply throttling to users who send the User-Agent header with the value 'Googlebot/2.1' and limits the number of API invocations to 5 requests per minute.

Advanced Rate Limiting Policy - Create new

General Details

Provide name and description of the policy. The policy can be referred from the name.

Name*

Name of the throttle policy.

Description

Description of the throttle policy.

Default Limits

Request Count and Request Bandwidth are the two options for default limit. You can use the option according to your requirement.

Default Limit Option ☒ Request Count ☐ Request Bandwidth

Request Count

Number of requests allowed

Unit Time Minute(s) ▼

Time configuration

Conditional groups

To add throttling limits with different parameters base on IP, Header, Query Param, and JWT Claim conditions, click Add Conditional Group.

Add Conditional Group

Hide group ^

Condition Policies

IP Condition Policy

This configuration is used to throttle by IP address.

Add

Header Condition Policy

This configuration is used to throttle based on Headers.

Add

Query Param Condition Policy

This configuration is used to throttle based on query parameters.

Add

JWT Condition Policy

This configuration is used to define JWT claims conditions

Add

Default Limit Option ☒ Request Count ☐ Request Bandwidth

Request Count

Number of requests allowed

Unit Time Minute(s) ▼

Time configuration

Description

Description of this group

Add

Cancel

Header Condition Policy
This configuration is used to throttle based on Headers.

Edit Header Condition Policy

This configuration is used to throttle based on Headers.

Header Name*
User-Agent

Provide Name

Header Value*
Googlebot/2.1

Provide Value

Cancel Save

Conditional groups
To add throttling limits with different parameters base on IP, Header, Query Param, and JWT Claim conditions, click Add Conditional Group.



Add Conditional Group

Hide group ^

Condition Policies

IP Condition Policy
This configuration is used to throttle by IP address. Add

Header Condition Policy
This configuration is used to throttle based on Headers. ☐ Invert Condition Add

| Header Name | Header Value | |
|-------------|---------------|---|
| User-Agent | Googlebot/2.1 |   |

Query Param Condition Policy
This configuration is used to throttle based on query parameters. Add

JWT Condition Policy
This configuration is used to define JWT claims conditions. Add

Default Limit Option ☒ Request Count ☐ Request Bandwidth

Request Count
5
Number of requests allowed

Unit Time
1
Time configuration

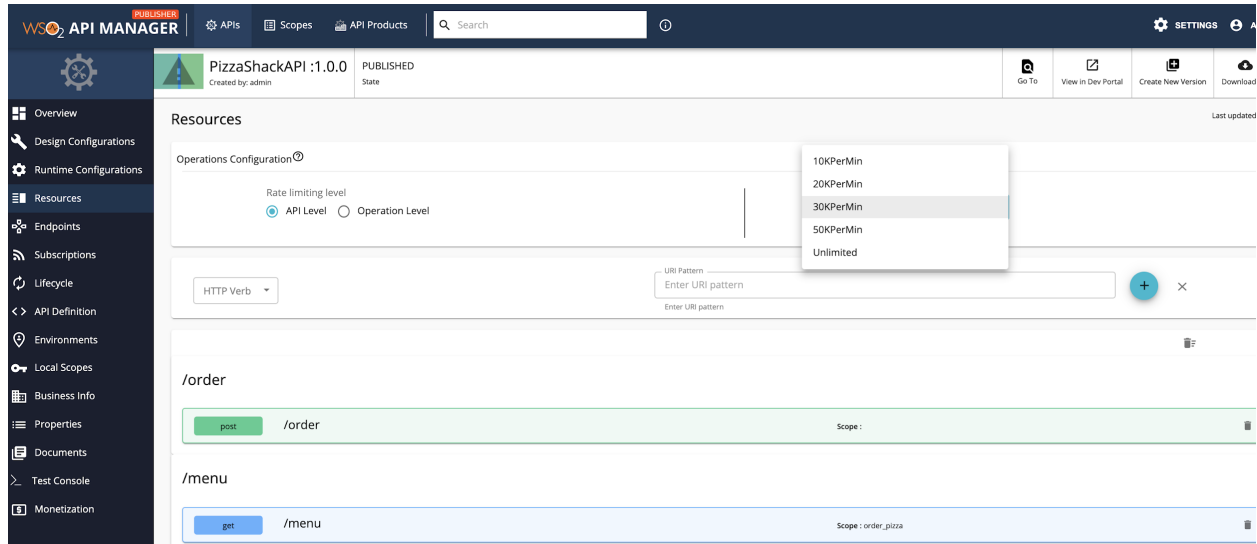
Minute(s) ▼

Description
Sample description about condition group
Description of this group

Add Cancel

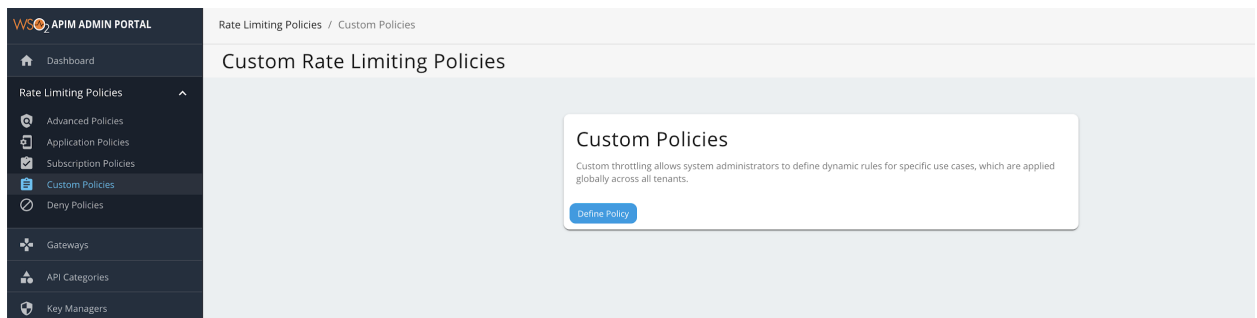
3. Click **Add** to add the Header condition Policy and Click **Add**.

The new advanced throttling policy will be available under the **Operation Configuration** for API in the **Resources** tab.



Add Custom Rules

1. Click **CUSTOM POLICIES** and select **DEFINE POLICY**.



2. Enter the following information

Custom Rate Limiting Policy - Define Policy

Name *
CustomRule

Name of the throttle policy


Description
Allow 5 requests per minute for the admin user

Description of the throttle policy

Key Template *
\$userId

The specific combination of attributes being checked in the policy need to be defined as the key template
Eg: \$userId, \$apiContext, \$apiVersion, \$resourceKey, \$appTenant, \$apiTenant, \$appId, \$clientip

Siddhi Query:

 The following sample query will allow 5 requests per minute for an Admin user.
Key Template : \$userId

```

1 FROM
2   RequestStream
3 SELECT
4   userId,
5   (userId == 'admin@carbon.super') AS isEligible,
6   str :concat('admin@carbon.super', '') as throttleKey
7 INSERT INTO
8   EligibilityStream;
9 FROM
10  EligibilityStream [isEligible==true] #throttler:timeBatch(1 min) SELECT throttleKey, (count(userId) >= 5) as isThrottled,

```

Add Cancel

Add the following details.

| | |
|---------------------|--|
| Name | CustomRule |
| Description | Allow 5 requests per minute for the admin user |
| Key Template | \$userId |

3. Click **Add**.

Blocking/Denying an Application

1. Log in to API Publisher and go to the edit view of **PizzaShack** API.
2. Click the **Resources** tab and make sure the **Operation Level** is selected under the **Operations Configuration**.

The screenshot shows the API Publisher interface for the PizzaShack API. The top navigation bar includes a 'BACK TO APIs' link, the API name 'PizzaShack :1.0.0', its state 'PUBLISHED', and buttons for 'Go To', 'View in Dev Portal', and 'Create'. The main section is titled 'Resources' and contains an 'Operations Configuration' tab. Under this tab, there are two radio buttons for 'Rate limiting level': 'API Level' and 'Operation Level'. The 'Operation Level' is selected. To the right of these buttons, there is a text box that says 'You may change the rate limiting policies per operation' and 'Expand an operation below to select a rate limiting policy for an operation'. Below this, there is a 'URI Pattern' input field with a placeholder 'Enter the URI pattern' and a 'HTTP Verb' dropdown menu.

3. Click **Save & Publish** if you have made changes to the API.
4. Go to API Devportal (<https://localhost:9443/devportal/>) and click **Register Now** in the Login Screen.
5. Self Signup a user with adding details as follows. (Alternatively, you can use a user which you have created before using the Sign Up option).

CREATE NEW ACCOUNT

Fill in the form below to complete registration

First Name *

David

Last Name *

Robinson

Password *

.....

Confirm password *

.....

Email *

david.robinson@gmail.com

Organization

Telephone

IM

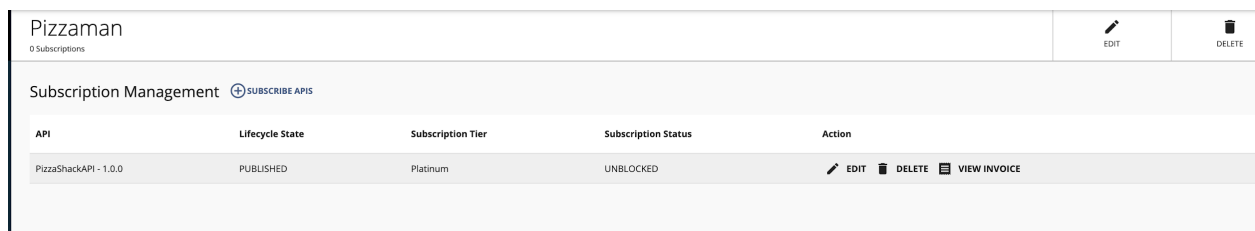
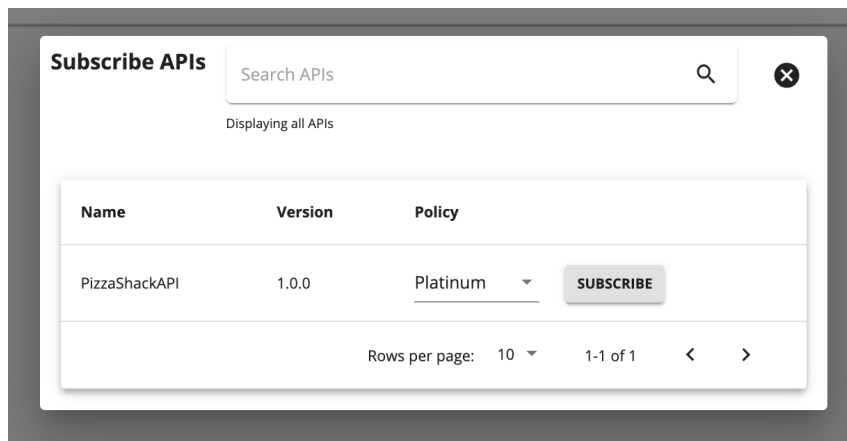
Country

Mobile

URL

When you sign in, we use a cookie in your browser to track your session. You can read our [Cookie Policy](#) for more information.

6. Log in to the API Devportal with the last created user's credentials.
7. Select carbon.super tenant
8. Go to **APPLICATIONS** and click **ADD NEW APPLICATION** on the top.
9. Create an Application named **"Pizzaman"**.
10. Subscribe to the **"PizzaShack"** API through **"Pizzaman"** with the **"Platinum"** tier.



11. Generate keys for the application under the Production keys tab.
12. Go to the API **Console** of **PizzaShackAPI**. Send a GET request to the Resource “menu” using the tryout tool. *

Repeat step 12. It should now display the response.

The screenshot shows a REST client interface with the following details:

- Execute** button and **Clear** button.
- Responses** section:

 - Curl**: A long curl command for a GET request to `https://localhost:8243/pizzashack/1.0.0/menu` with headers for `Accept` and `Authorization: Bearer`.
 - Request URL**: `https://localhost:8243/pizzashack/1.0.0/menu`
 - Server response**:
 - Code**: 200
 - Details**:
 - Response body**: A JSON array of menu items. Each item has a name, description, price, and icon path. Items include "BBQ Chicken Bacon", "Chicken Parmesan", "Chilly Chicken Cordon Bleu", "Double Bacon 6 Cheese", and "Garden Fresh".
 - Response headers**: `content-type: application/json`

- Responses** table at the bottom:

| Code | Description | Links |
|------|-------------|----------|
| 200 | ok | No links |

Alternatively, you can use the following cURL/Rest API command

Replace the Authorization Bearer token with the access token retrieved when generating the Keys for **PizzaMan** Application.

```
curl -k -X GET --header 'Accept: application/json' --header 'Authorization: Bearer 31bdb476-6b28-360e-a61e-25845b4e4921' 'https://localhost:8243/pizzashack/1.0.0/menu'
```

Method: GET

URL: `http://localhost:8280/pizzashack/1.0.0/menu`

Authorization tab - Type: Bearer Token,

Token : “d919d61a-8ef4-3059-b28e-9f38023aa306”

Application PizzaMan can now successfully invoke the API.

13. Now login to the API Manager Admin Portal (<https://localhost:9443/admin/>) and click **DENY POLICIES** under **RATE LIMITING POLICIES**.

14. Click on **ADD POLICY**, select **Application** and add the application name with the username which needs to be blocked/denied in the following format.

<username>:<applicationName>, which is "**david:Pizzaman**"

Select Item to Deny

Condition Type

☐ API Context
 ☒ Application
 ☐ IP Address
 ☐ IP Range
 ☐ User

Value *

Format : \${userName}:\${applicationName}

Eg : admin:DefaultApplication

☒ Enable Condition

Cancel
 Deny

Denied Application will be listed as follows.

WSO₂ API ADMIN PORTAL

- Dashboard
- Rate Limiting Policies
 - Advanced Policies
 - Application Policies
 - Subscription Policies
 - Custom Policies
 - Deny Policies
- Gateways
- API Categories
- Key Managers

Rate Limiting Policies / Deny Policies

Deny Policies

Add Policy

| Condition Type | Conditional Value | Condition Status | Actions |
|----------------|---------------------|-------------------------------------|---------|
| Application | subscriber:Pizzaman | <input checked="" type="checkbox"/> | |

Rows per page: 10
 1-1 of 1

15. Now go to the Developer Portal again and invoke the **PizzaShackAPI** same as in step 11 and 12 using the **Pizzaman** application.

You should receive the following response.

```
{
  "fault": {
    "code": 900805,
    "message": "Message blocked",
    "description": "You have been blocked from accessing the
resource"
  }
}
```

The screenshot shows the WSO2 API Manager console interface. At the top, there are 'Execute' and 'Clear' buttons. Below, the 'Responses' section displays the details of a failed request.

Request URL: `https://localhost:8243/pizzashack/1.0.0/menu`

Server response:

| Code | Details |
|----------------------------|---|
| 403 <i>Undocumented</i> | <p>Error: Forbidden</p> <p>Response body</p> <pre><amt:fault xmlns:amt="http://ws2.org/apimanager/throttling"> <amt:code>900805</amt:code> <amt:message>Message blocked</amt:message> <amt:description>You have been blocked from accessing the resource</amt:description> </amt:fault></pre> <p>Response headers</p> <pre>content-type: application/xml; charset=UTF-8</pre> |

At the bottom, a 'Responses' table shows the status of the request:

| Code | Description | Links |
|------|-------------|----------|
| 200 | ok | No links |

Expected Outcome

Your new subscription tier (Platinum) is now successfully saved as an execution plan used by WSO2 API Manager. You can view this new rate limiting policy available for selection when creating a new API through the API Publisher or when editing an existing API.

Your new advanced rate limiting policy 30KPerMin, with conditional rate limiting groups, is now successfully saved as a rate limiting policy. You can apply it to the whole API or selected resources.



Invoking the PizzaShackAPI by the specific user (Subscriber) through Pizzaman API is now blocked as the application Pizzaman is denied.