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 is increasing so they have decided to allow up to 100 requests per minute. Other than that, in a recent analysis they found out that PizzaShack API is getting misused through an application called "Pizzaman" and they want to block all calls from that application.

High-Level Steps

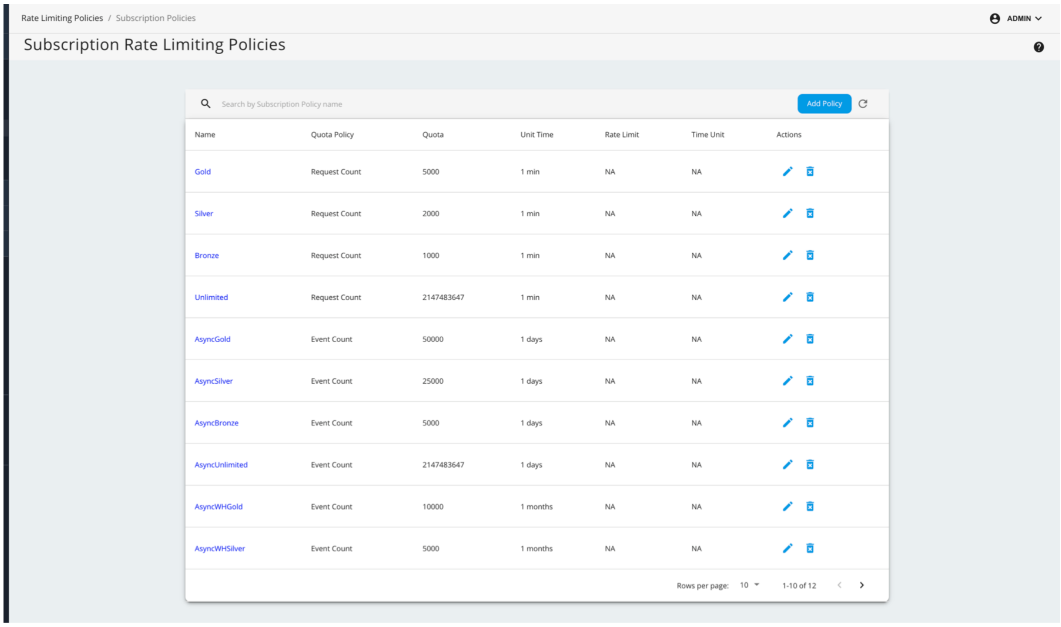
- Add rate limiting policy
- Add conditions to rate limiting policy
- Add custom rules
- Block all calls from an application by denying an application.

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** from the left menu select **Subscription Policies** and then select **Add Policy** at the top.

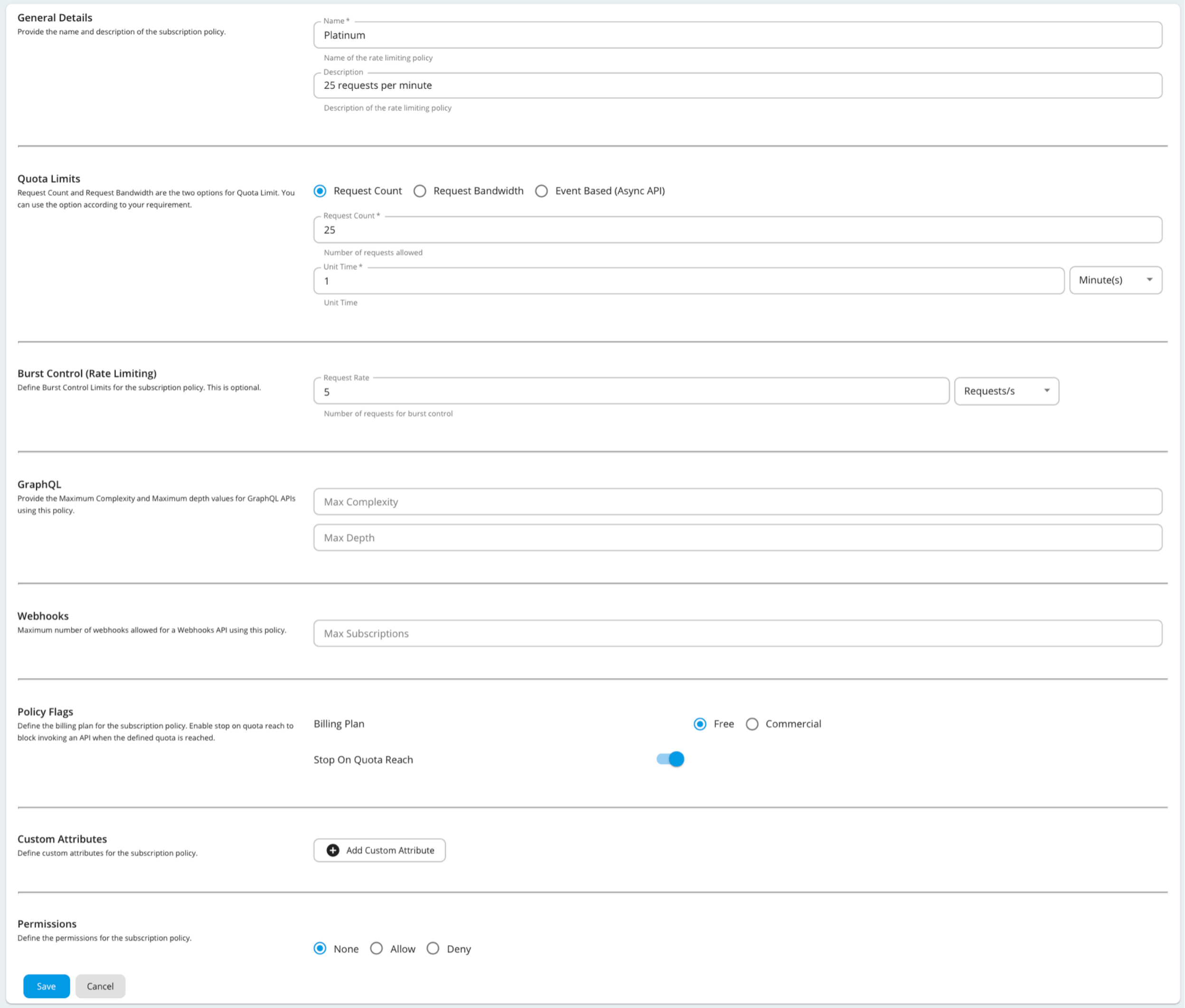


Rate Limiting Policies / Subscription Policies
ADMIN
Search by Subscription Policy name
Add Policy
Rows per page: 10
1-10 of 12

Subscription Rate Limiting Policies

Name	Quota Policy	Quota	Unit Time	Rate Limit	Time Unit	Actions
Gold	Request Count	5000	1 min	NA	NA	
Silver	Request Count	2000	1 min	NA	NA	
Bronze	Request Count	1000	1 min	NA	NA	
Unlimited	Request Count	2147483647	1 min	NA	NA	
AsyncGold	Event Count	50000	1 days	NA	NA	
AsyncSilver	Event Count	25000	1 days	NA	NA	
AsyncBronze	Event Count	5000	1 days	NA	NA	
AsyncUnlimited	Event Count	2147483647	1 days	NA	NA	
AsyncWHGold	Event Count	10000	1 months	NA	NA	
AsyncWHSilver	Event Count	5000	1 months	NA	NA	

3. Enter the following information and click **Save.**



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 ○ Event Based (Async API)

Request Count * : 25

Number of requests allowed

Unit Time * : 1 Minute(s)

Unit Time

Burst Control (Rate Limiting)

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

Request Rate: 5 Requests/s

Number of requests for burst control

GraphQL

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

Max Complexity

Max Depth

Webhooks

Maximum number of webhooks allowed for a Webhooks API using this policy.

Max Subscriptions

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: on

Custom Attributes

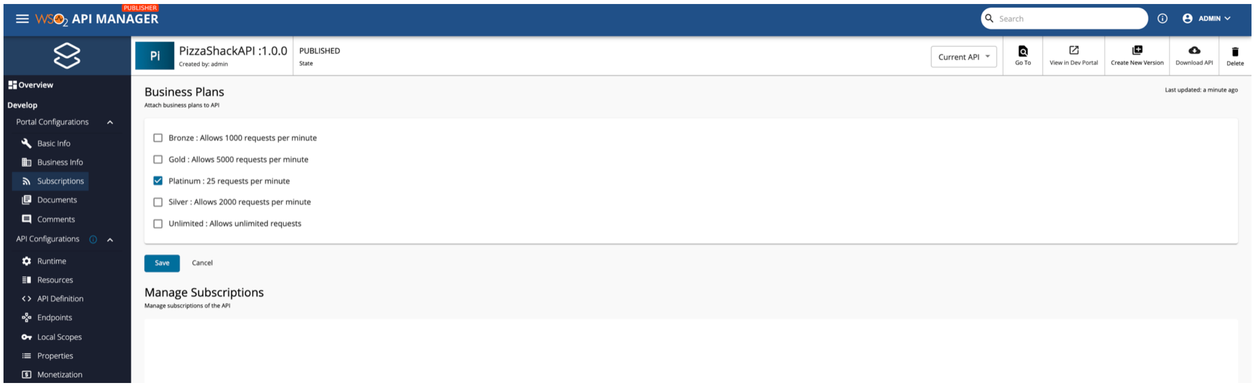
Define custom attributes for the subscription policy.

Permissions

Define the permissions for the subscription policy.

● None ○ Allow ○ Deny

In the API **Publisher**, edit the PizzaShack API and note that **Platinum** can now be selected as a Business Plan for the API. **Tick the checkbox of the Platinum tier** under **Business Plans** in the **Portal Configurations > Subscriptions** tab and save it. The API will be visible in the API Developer Portal whenever a person wants to subscribe to the PizzaShack API.



PI PizzaShackAPI :1.0.0
Created by: admin

PUBLISHED
State

Business Plans
Attach business plans to API

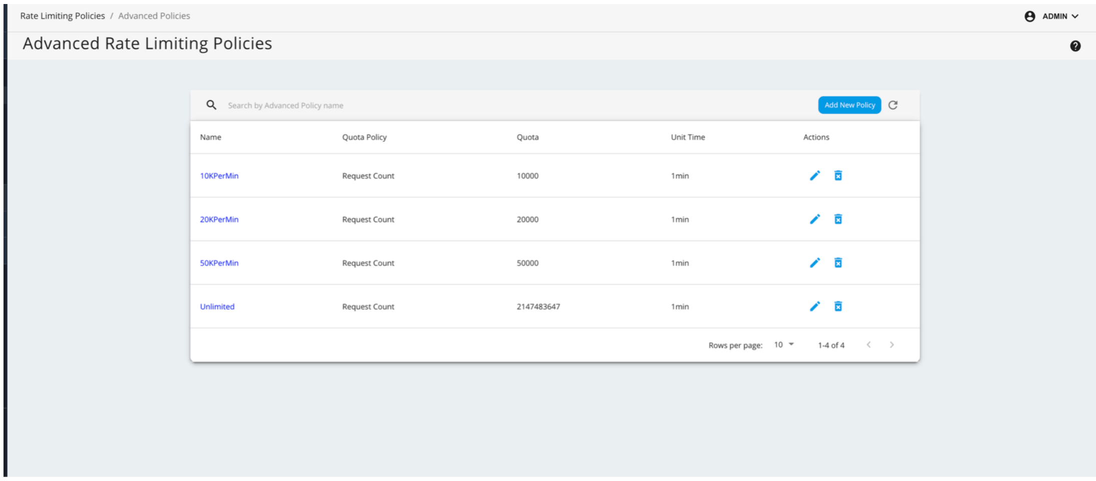
Last updated: a minute ago

- □ Bronze : Allows 1000 requests per minute
- □ Gold : Allows 5000 requests per minute
- ☑ Platinum : 25 requests per minute
- □ Silver : Allows 2000 requests per minute
- □ Unlimited : Allows unlimited requests

Manage Subscriptions
Manage subscriptions of the API

Add Conditions to Advanced Rate Limiting

1. Click **Advanced Policies** and select **Add New Policy** at the top.



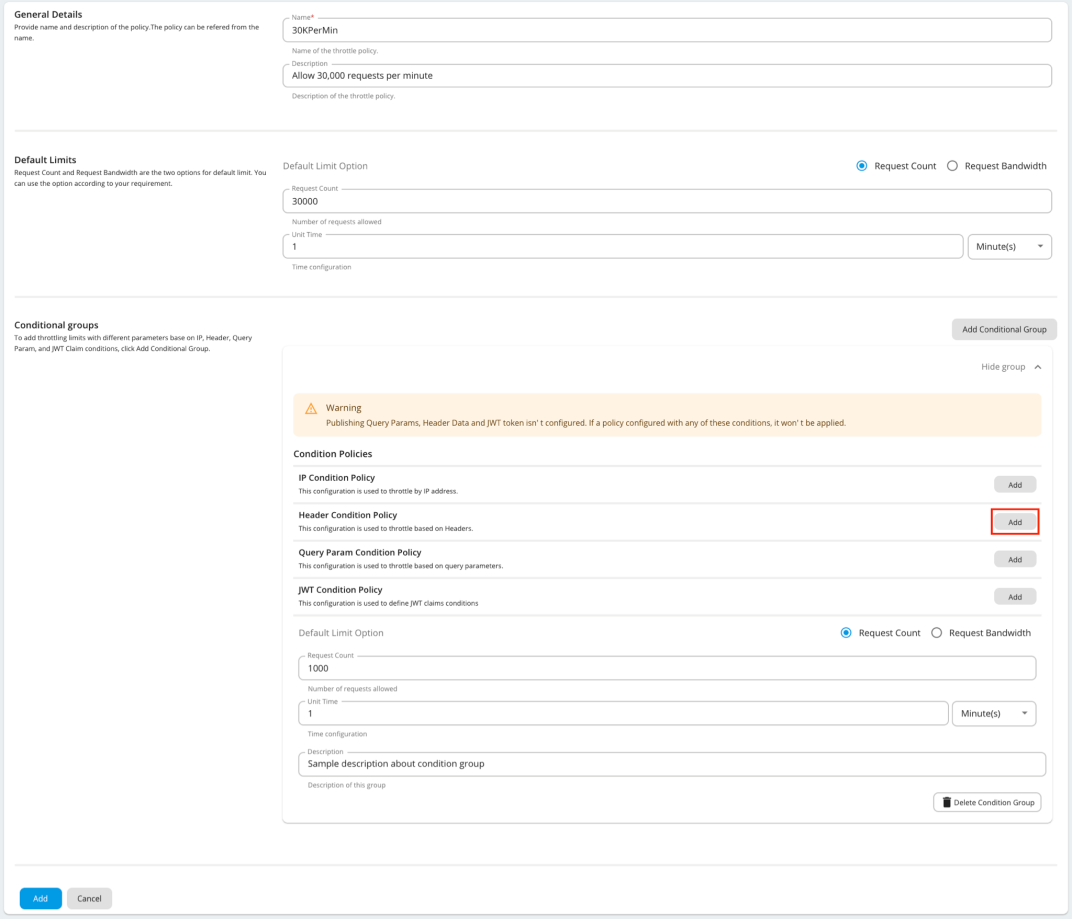
Rate Limiting Policies / Advanced Policies
ADMIN
Search by Advanced Policy name
Add New Policy

Advanced Rate Limiting Policies

Name	Quota Policy	Quota	Unit Time	Actions
10KPerMin	Request Count	10000	1min	
20KPerMin	Request Count	20000	1min	
50KPerMin	Request Count	50000	1min	
Unlimited	Request Count	2147483647	1min	

Rows per page: 10 1-4 of 4

2. Enter the following information and click **Add Conditional Group.** Expand to edit and select **Header Condition Policy** and add the details as shown in the images.



General Details

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

Name*: 30KPerMin

Name of the throttle policy.

Description: Allow 30,000 requests per minute

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: 30000

Number of requests allowed

Unit Time: 1 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.

Warning

Publishing Query Params, Header Data and JWT token isn' t configured. If a policy configured with any of these conditions, it won' t be applied.

Condition Policies

IP Condition Policy

This configuration is used to throttle by IP address.

Header Condition Policy

This configuration is used to throttle based on Headers.

Query Param Condition Policy

This configuration is used to throttle based on query parameters.

JWT Condition Policy

This configuration is used to define JWT claims conditions

Default Limit Option ● Request Count ○ Request Bandwidth

Request Count: 1000

Number of requests allowed

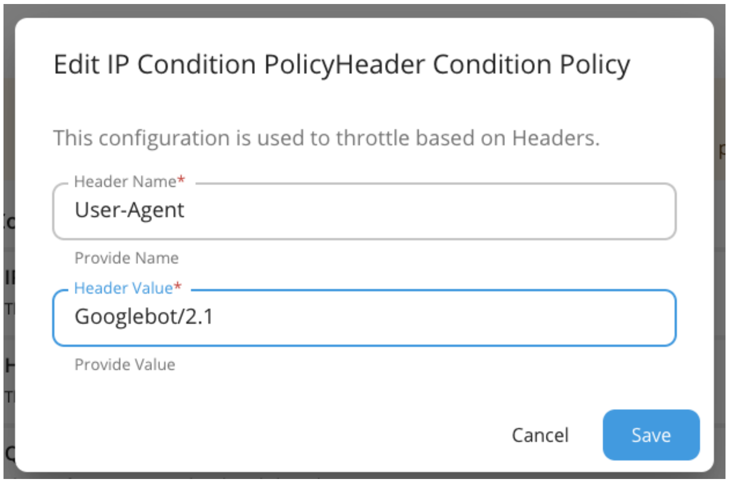
Unit Time: 1 Minute(s)

Time configuration

Description: Sample description about condition group

Description of this group

3. 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 1000 requests per minute.



Edit IP Condition PolicyHeader Condition Policy

This configuration is used to throttle based on Headers.

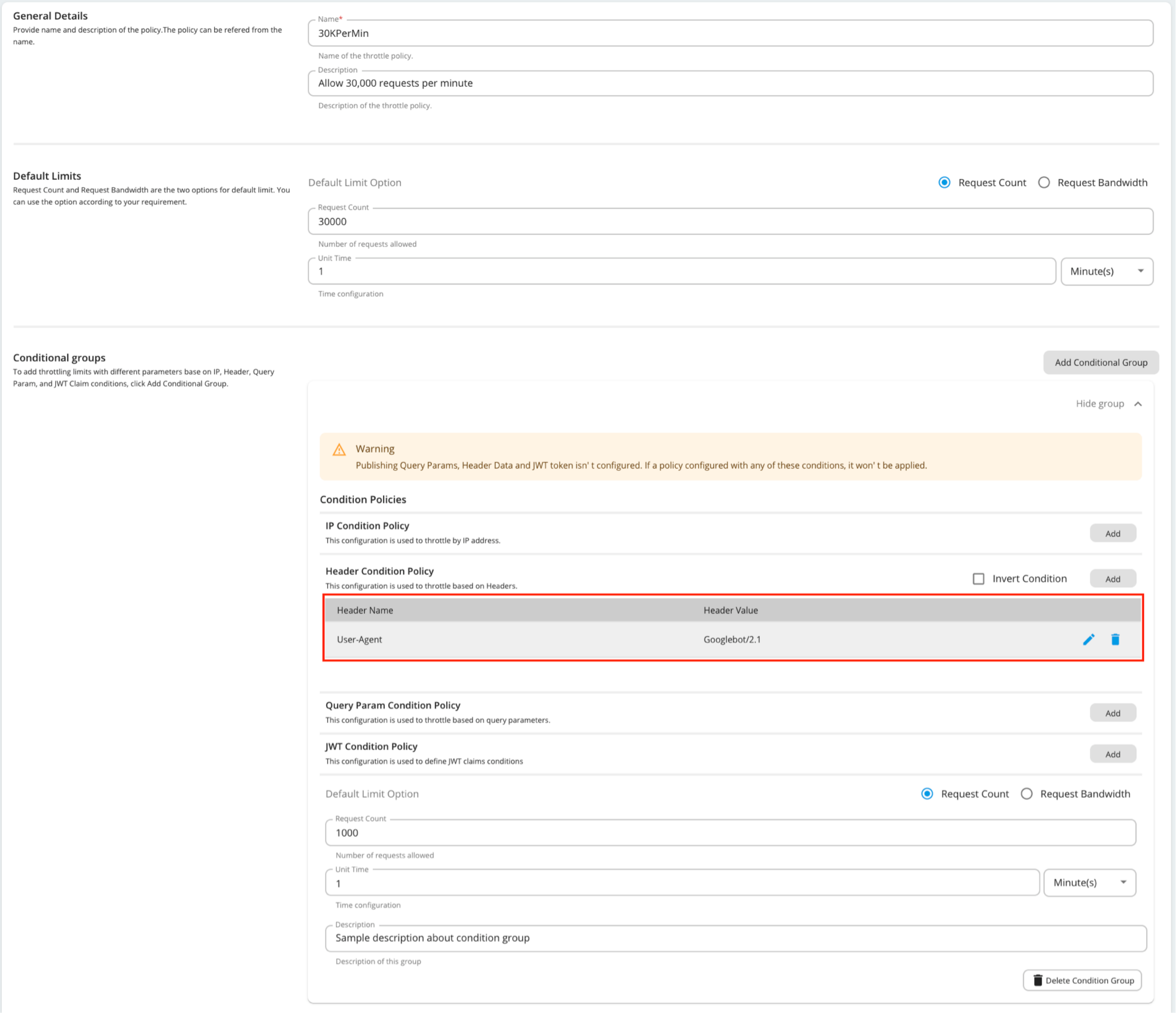
Header Name*: User-Agent

Provide Name

Header Value*: Googlebot/2.1

Provide Value

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



General Details
Provide name and description of the policy.The policy can be refered from the name.

Name*: 30KPerMin
Name of the throttle policy.

Description: Allow 30,000 requests per minute
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: 30000
Number of requests allowed

Unit Time: 1 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.

Warning
Publishing Query Params, Header Data and JWT token isn' t configured. If a policy configured with any of these conditions, it won' t be applied.

Condition Policies

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

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

Header Name	Header Value
User-Agent	Googlebot/2.1

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

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

Default Limit Option ● Request Count ○ Request Bandwidth

Request Count: 1000
Number of requests allowed

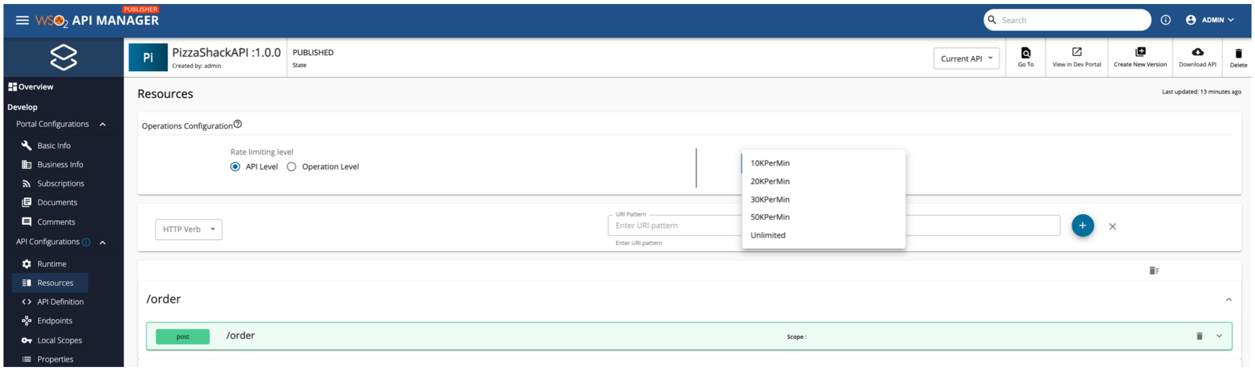
Unit Time: 1 Minute(s)
Time configuration

Description: Sample description about condition group
Description of this group

5. Note that you need to enable the header condition throttling specifically by adding the `enable_header_based_throttling` configuration below into the *deployment.toml* file available under <API-M_HOME>/repository/conf/.

```
[apim.throttling]
enable_header_based_throttling = true
```

The new advanced throttling policy will be available under the **Operations Configuration** for API in the **Resources** tab in the API Publisher: https://localhost:9443/publisher



PizzaShackAPI :1.0.0
Created by: admin
PUBLISHED
State

Resources

Last updated: 13 minutes ago

Operations Configuration

Rate limiting level
● API Level ○ Operation Level

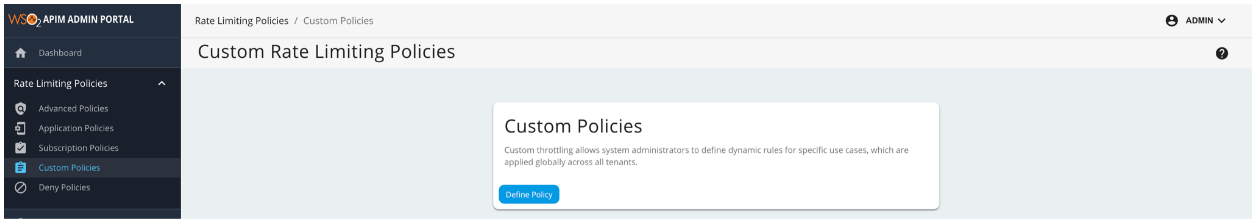
10KPerMin
20KPerMin
30KPerMin
50KPerMin
Unlimited

/order

post /order Scope :

Add Custom Rules

1. Click **Custom Policies** and select **Define Policy**.



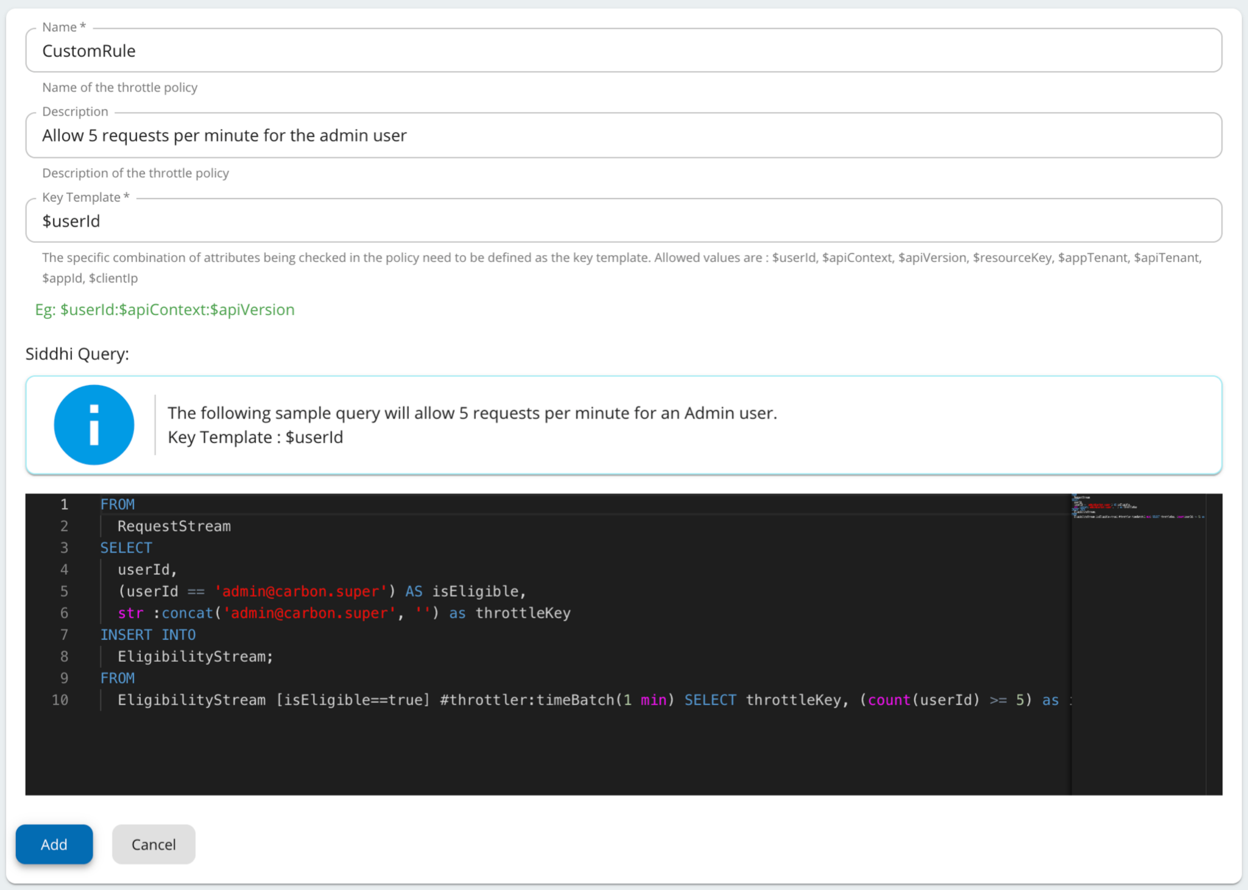
Rate Limiting Policies / Custom Policies

Custom Rate Limiting Policies

Custom Policies

Custom throttling allows system administrators to define dynamic rules for specific use cases, which are applied globally across all tenants.

2. Enter the following information



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. Allowed values are : $userId, $apiContext, $apiVersion, $resourceKey, $appTenant, $apiTenant, $appId, $clientIp

Eg: $userId:$apiContext:$apiVersion

Siddhi Query:

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

```
FROM
  RequestStream
SELECT
  userId,
  (userId == 'admin@carbon.super') AS isEligible,
  str :concat('admin@carbon.super', '') as throttleKey
INSERT INTO
  EligibilityStream;
FROM
  EligibilityStream [isEligible==true] #throttler:timeBatch(1 min) SELECT throttleKey, (count(userId) >= 5) as
```

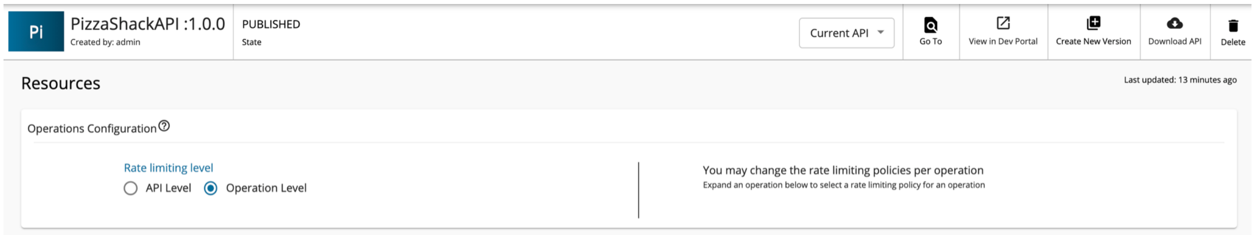
Add the following details.

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**.



PizzaShackAPI :1.0.0
Created by: admin

PUBLISHED
State

Resources

Last updated: 13 minutes ago

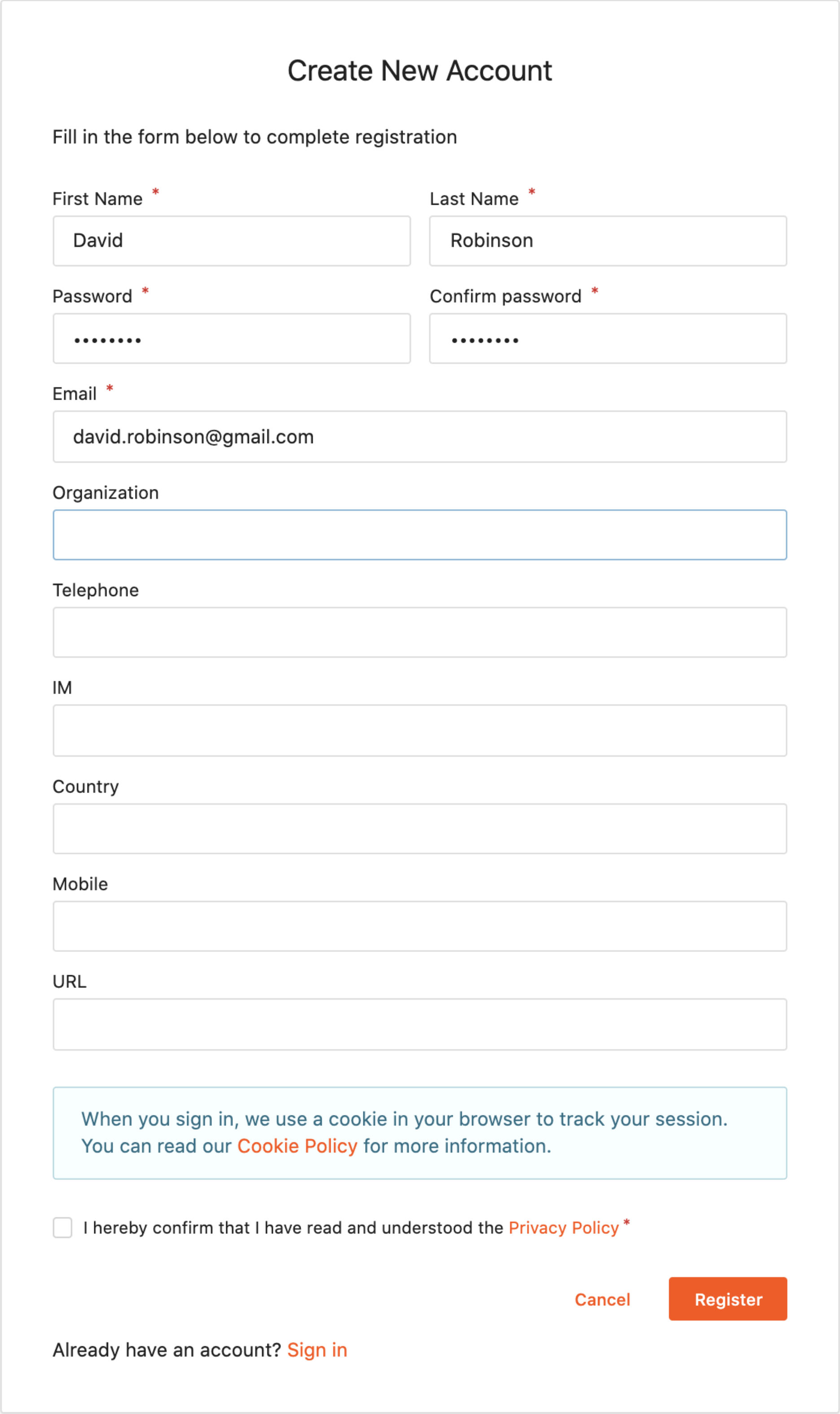
Operations Configuration

Rate limiting level
○ API Level ● Operation Level

You may change the rate limiting policies per operation
Expand an operation below to select a rate limiting policy for an operation

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

WSO2



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

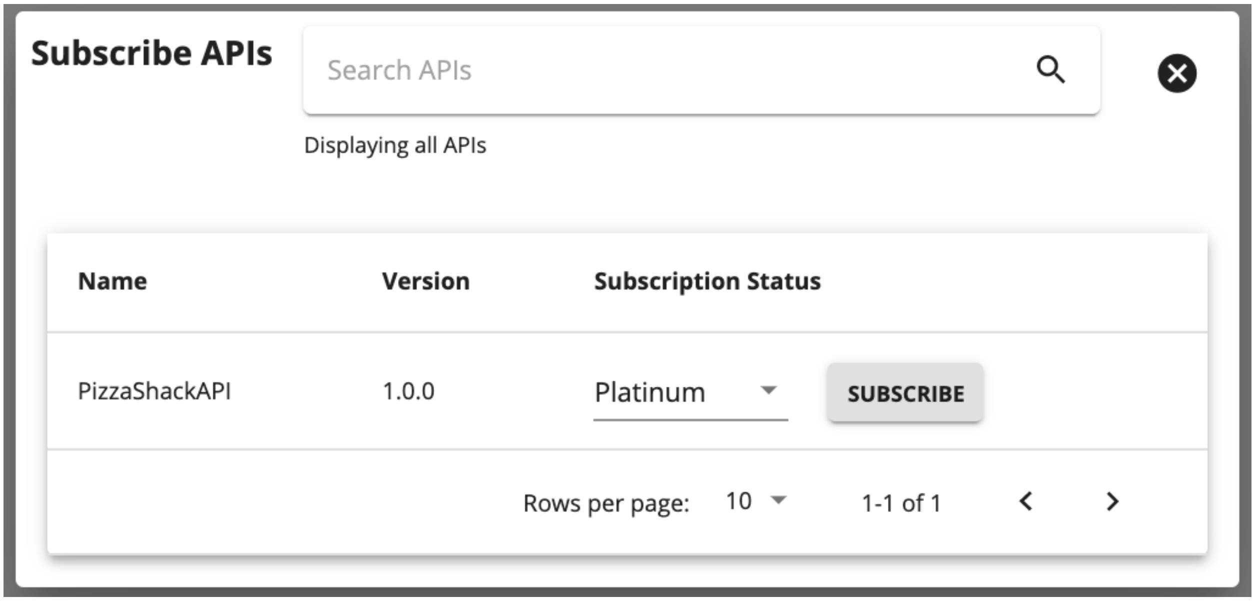
URL

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

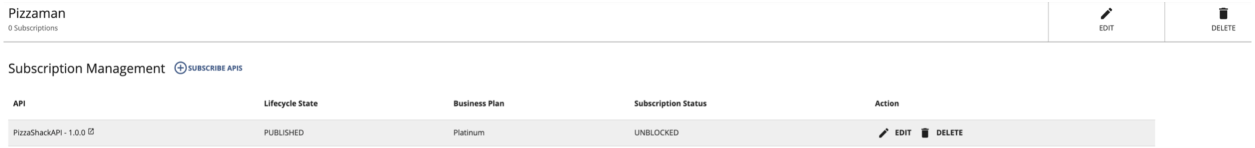
□ I hereby confirm that I have read and understood the Privacy Policy *

Already have an account? Sign in

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



Name	Version	Subscription Status
PizzaShackAPI	1.0.0	Platinum

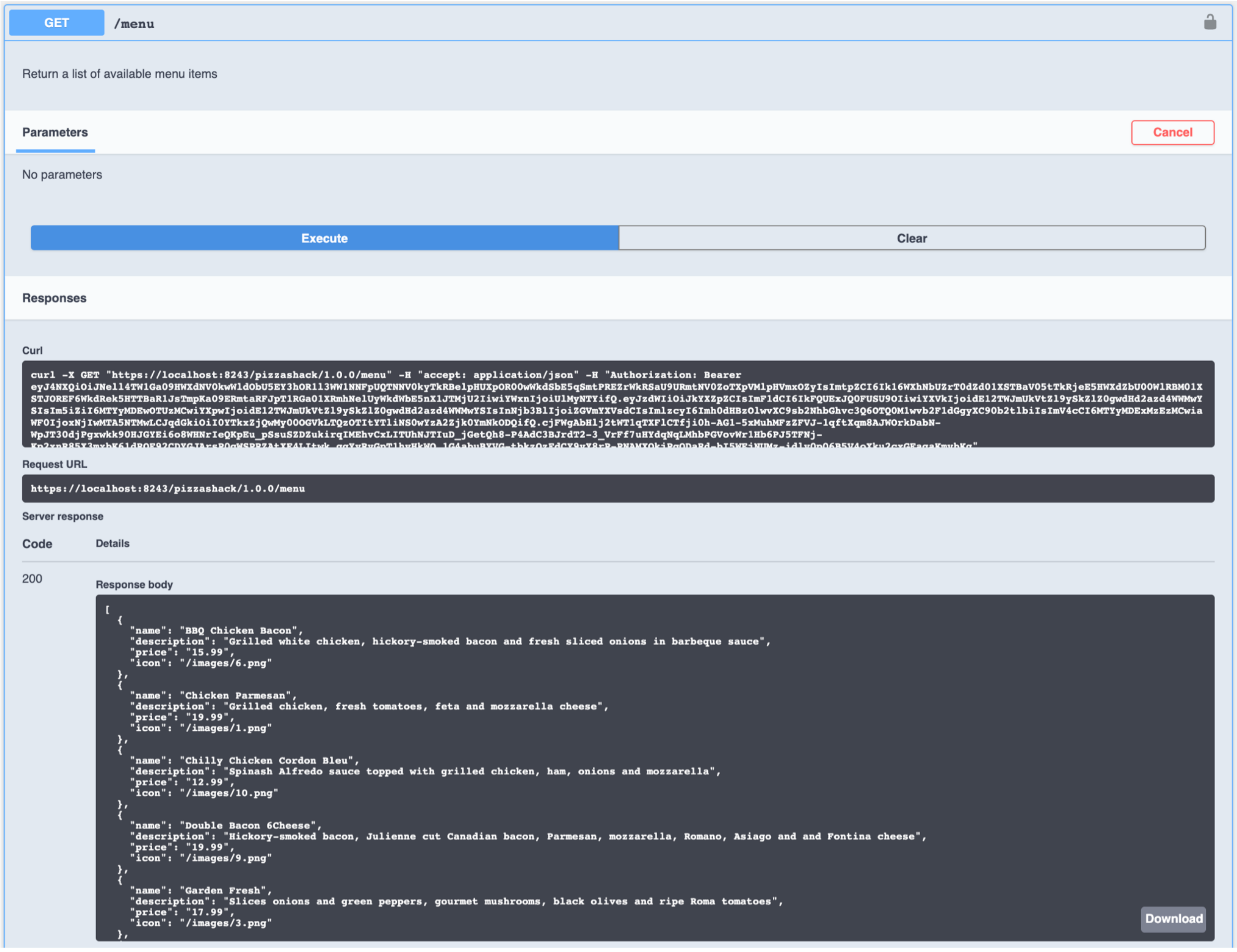


Pizzaman
0 Subscriptions

Subscription Management

API	Lifecycle State	Business Plan	Subscription Status	Action
PizzaShackAPI - 1.0.0	PUBLISHED	Platinum	UNBLOCKED	EDIT DELETE

11. Generate OAuth2 keys for the application under the Production keys tab.
12. Go to the API **Try Out** console of **PizzaShack** API**.** Send a GET request to the Resource "**menu**" using the tryout tool. It should now display the response.



Return a list of available menu items

Parameters

No parameters

Responses

Curl

```
curl -X GET "https://localhost:8243/pizzashack/1.0.0/menu" -H "accept: application/json" -H "Authorization: Bearer
eyJ4NXQiOiJNell4TW1Ga09HWXdNV0kwW1dObU5EY3hOR1l3WW1NNFpUQTNNV0kyTkRBelpHUXpOR0OwWkdSbE5qSmtPREZrWkRSaU9URmtNV0ZoTXpVMlpHVmxOZyIsImtpZCI6Ik16WXhNbUZrT0dZd01XSTBaV05tTkRjeE5HWXdZbU00WlRBM01X
STJOREF6WkdRek5HTTBaR1JsTmpKa09ERmtaRFJpT1RGa01XRmhNelUyWkdWbE5nX1JTMjU2IiwiYWxnIjoiUlMyNTYifQ.eyJzdWIiOiJkYXZpZCIsImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoidE12TWJmUkVtZl9ySkZlZ0gwdHd2azd4WWMwY
SIsIm5iZiI6MTYyMDEwOTUzMCwiYXpwIjoidE12TWJmUkVtZl9ySkZlZ0gwdHd2azd4WWMwYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0dHBzOlwvXC9sb2NhbGhvc3Q6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6MTYyMDExMzEzMCwia
WF0IjoxNjIwMTA5NTMwLCJqdGkiOiI0YTkxZjQwMy00OGVkLTQzOTItYTliNS0wYzA2Zjk0YmNkODQifQ.cjFWgAbHlj2tWT1qTXFlCTfji0h-AG1-5xMuhMFzZFVJ-1qftXqm8AJWOrkDabN-
WpJT30djPgxwkk90HJGYEi6o8WHNrIeQKpEu_pSsuSZDZukirqIMEhvCxLITUhNJTIuD_jGetQh8-P4AdC3BJrdT2-3_VrFf7uHYdqNqLMhbPGVovWr1Hb6PJ5TFNj-
Kp2xnR85X3mxbK61dROE92CDYGJArsR0qWSPRZAtXF4LJtwk_qqYvRyGnTlhyHkWO_1G4abuBYVG-tbkzOzFdCY9vY8rR-RNAMXOkiRgODaRd-bI5WFjNUMz-idlv0n06B5V4oXku2cxGEaqaKmvbKg"
```

Request URL

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

Server response

Code Details

200

Response body

```
[
  {
    "name": "BBQ Chicken Bacon",
    "description": "Grilled white chicken, hickory-smoked bacon and fresh sliced onions in barbeque sauce",
    "price": "15.99",
    "icon": "/images/6.png"
  },
  {
    "name": "Chicken Parmesan",
    "description": "Grilled chicken, fresh tomatoes, feta and mozzarella cheese",
    "price": "19.99",
    "icon": "/images/1.png"
  },
  {
    "name": "Chilly Chicken Cordon Bleu",
    "description": "Spinash Alfredo sauce topped with grilled chicken, ham, onions and mozzarella",
    "price": "12.99",
    "icon": "/images/10.png"
  },
  {
    "name": "Double Bacon 6Cheese",
    "description": "Hickory-smoked bacon, Julienne cut Canadian bacon, Parmesan, mozzarella, Romano, Asiago and and Fontina cheese",
    "price": "19.99",
    "icon": "/images/9.png"
  },
  {
    "name": "Garden Fresh",
    "description": "Slices onions and green peppers, gourmet mushrooms, black olives and ripe Roma tomatoes",
    "price": "17.99",
    "icon": "/images/3.png"
  },
```

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

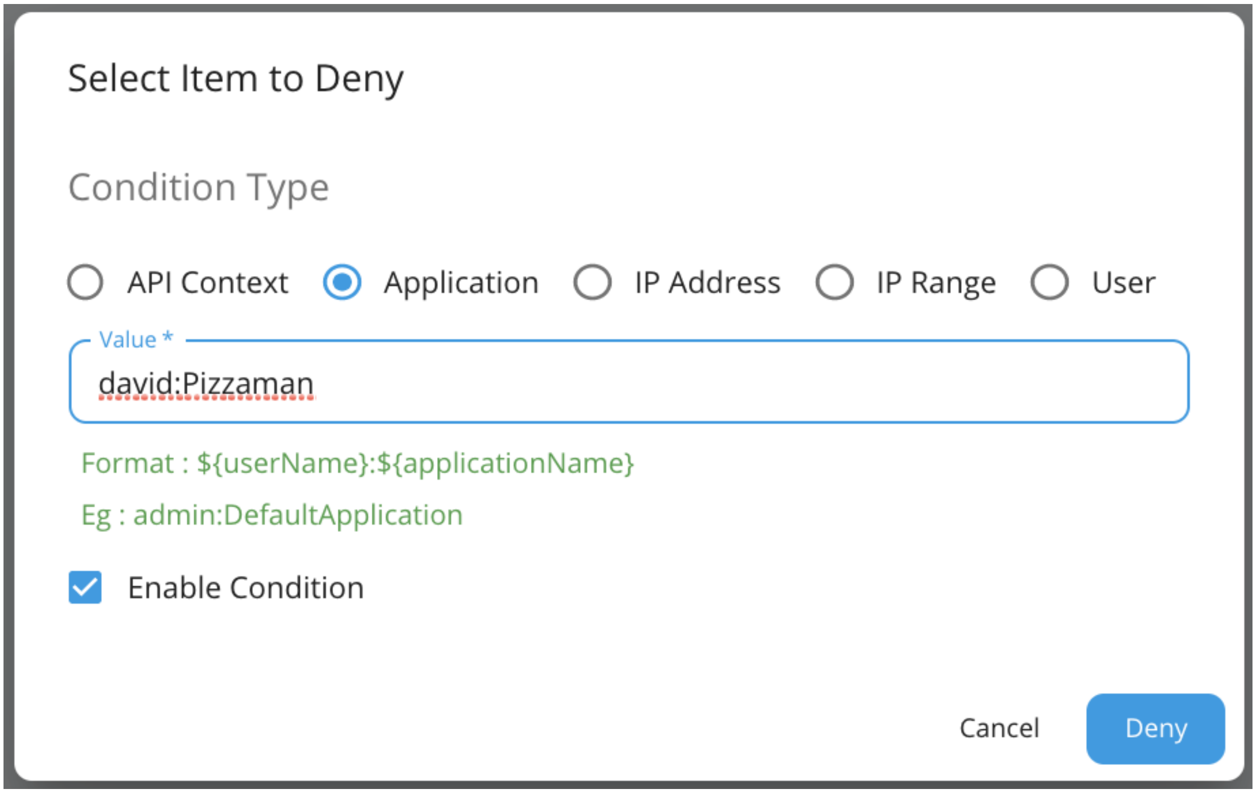
Replace the Authorization Bearer token with the access token retrieved when generating the Keys for **Pizzaman** Application.

```
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 *

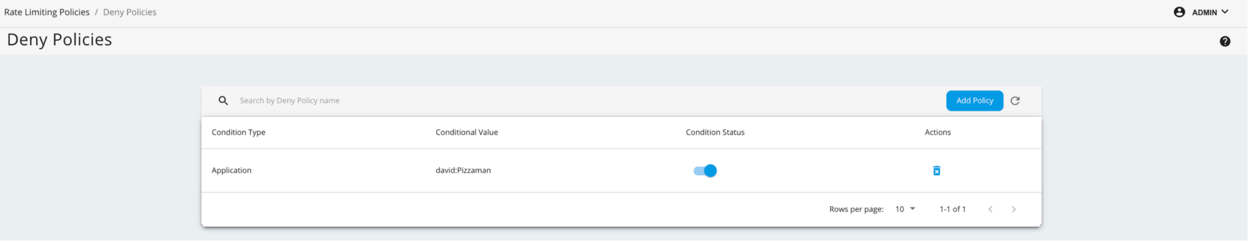
david:Pizzaman

Format : ${userName}:${applicationName}

Eg : admin:DefaultApplication

☑ Enable Condition

Denied Application will be listed as follows.



Rate Limiting Policies / Deny Policies
ADMIN

Deny Policies

Search by Deny Policy name
Add Policy

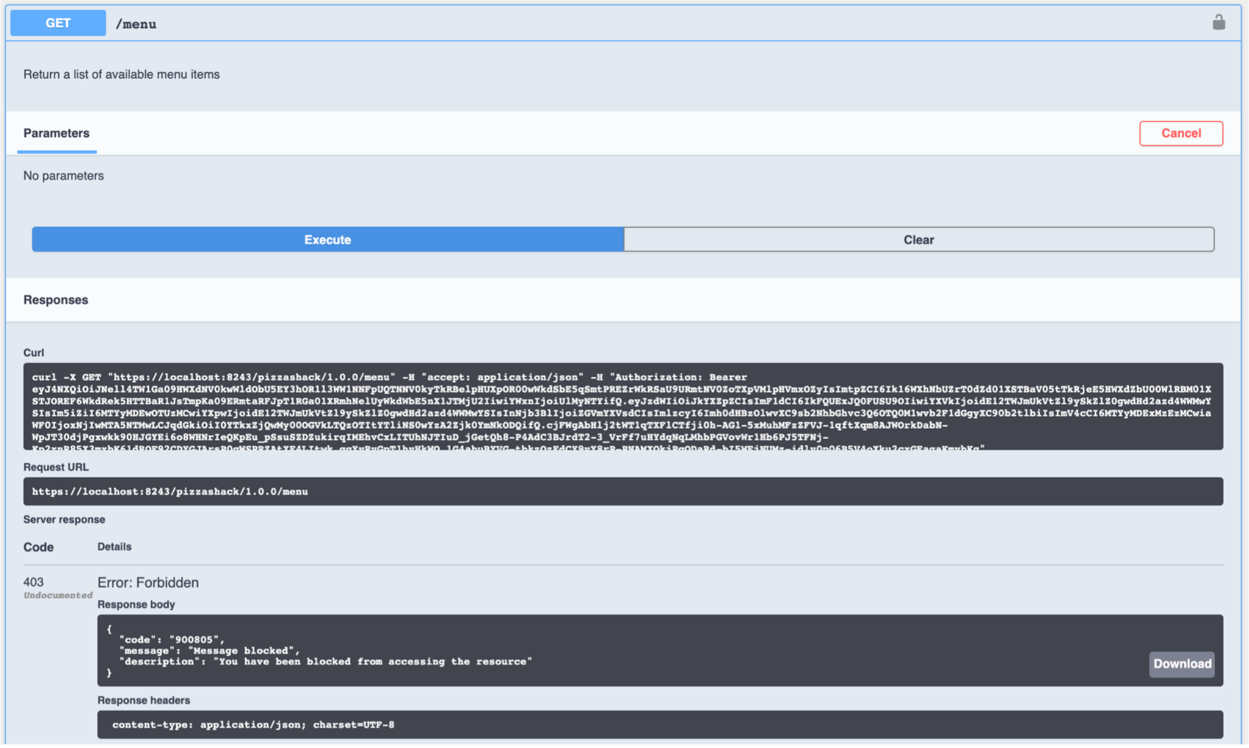
Condition Type	Conditional Value	Condition Status	Actions
Application	david:Pizzaman	●	🗑

Rows per page: 10 1-1 of 1

15. Now go to the Developer Portal again and invoke the **PizzaShack** API 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"
  }
}
```



Return a list of available menu items

No parameters

```
curl -X GET "https://localhost:8243/pizzashack/1.0.0/menu" -H "accept: application/json" -H "Authorization: Bearer
eyJ4NXQiOiJNell4TW1Ga09HWXdNV0kwWldObU5EY3hOR1l3WW1NNFpUQTNNV0kyTkRBelpHUXpOR00wWkdSbE5qSmtPREZrWkRSaU9URmtNV0ZoTXpVMlpHVmxOZyIsImtpZCI6Ik16WXhNbUZrT0dZd01XSTBaV05tTkRjeE5HWXdZbU00WlRBM01X
STJOREF6WkdRek5HTTBaR1JsTmpKa09ERmtaRFJpT1RGa01XRmhNelUyWkdWbE5nX1JTMjU2IiwiYWxnIjoiUlMyNTYifQ.eyJzdWIiOiJkYXZpZCIsImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoidE12TWJmUkVtZl9ySkZlZ0gwdHd2azd4WWMwY
SIsIm5iZiI6MTYyMDEwOTUzMCwiYXpwIjoidE12TWJmUkVtZl9ySkZlZ0gwdHd2azd4WWMwYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0dHBzOlwvXC9sb2NhbGhvc3Q6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6MTYyMDExMzEzMCwia
WF0IjoxNjIwMTA5NTMwLCJqdGkiOiI0YTkxZjQwMy00OGVkLTQzOTItYTliNS0wYzA2Zjk0YmNkODQifQ.cjFWgAbHlj2tWT1qTXFlCTfji0h-AG1-5xMuhMFzZFVJ-1qftXqm8AJWOrkDabN-
WpJT30djPgxwkk90HJGYEi6o8WHNrIeQKpEu_pSsuSZDZukirqIMEhvCxLITUhNJTIuD_jGetQh8-P4AdC3BJrdT2-3_VrFf7uHYdqNqLMhbPGVovWr1Hb6PJ5TFNj-
```

Request URL

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

Server response

Code	Details
403 *Undocumented*	Error: Forbidden

Response body

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

Response headers

```
 content-type: application/json; charset=UTF-8
```

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 PizzaShack API by the specific user (a subscriber) through Pizzaman API is now blocked as the application Pizzaman is denied.