

# How to Configure the REST API Call Bot Block

The API write-back can be customized by configuring the REST API Call bot block. Customizing the API write-back and creating an API REST block enables MSPbots to establish a seamless integration with your systems and allows us to retrieve relevant data so we can create reports that will help you monitor your performance, receive alerts, and perform necessary actions programmatically.

This article is a guide on how to configure the REST API Call bot block. It will use the **Monitoring [Opened Tickets Today - Template] for \_test** bot as an example.

What's on this page:

- [Prerequisites](#)
- [Required Permissions](#)
- [To set up the bot block](#)
- [Examples of commonly used HTTP Methods when using the REST API Call block](#)
- [Related topics](#)

## Prerequisites

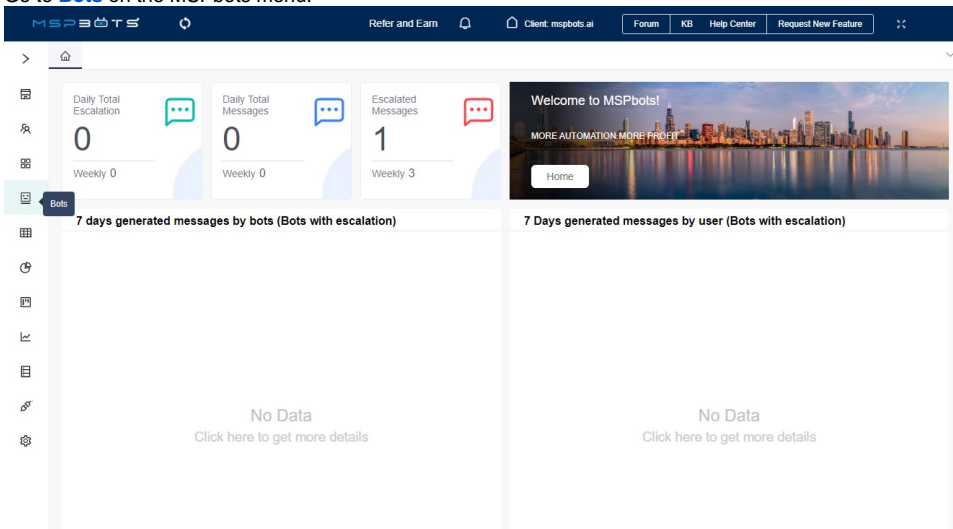
1. A bot of the trigger type has been created.
2. A widget or dataset has been created to filter integration data according to your needs.
3. The API documentation for the corresponding integration has been found, and APIs that meet your requirements have been identified.

## Required Permissions

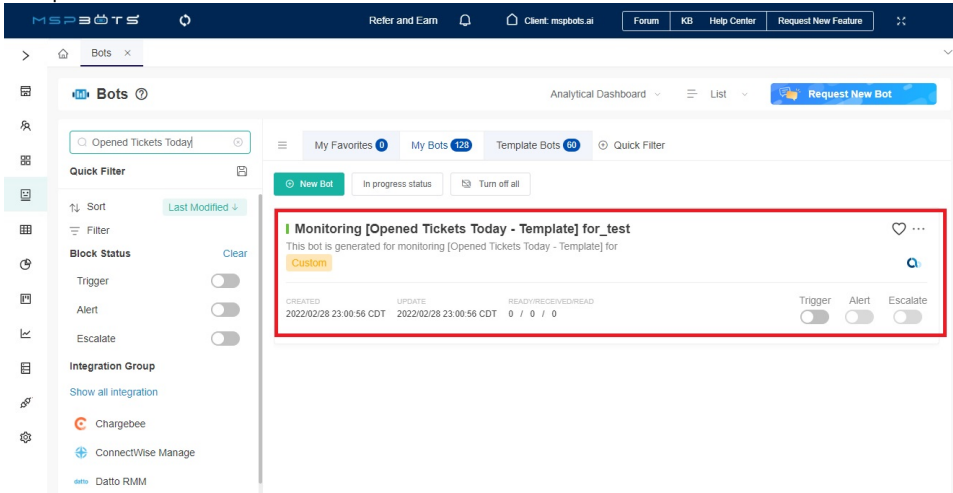
To follow this guide, you need admin-level permissions to perform the operations.

## To set up the bot block

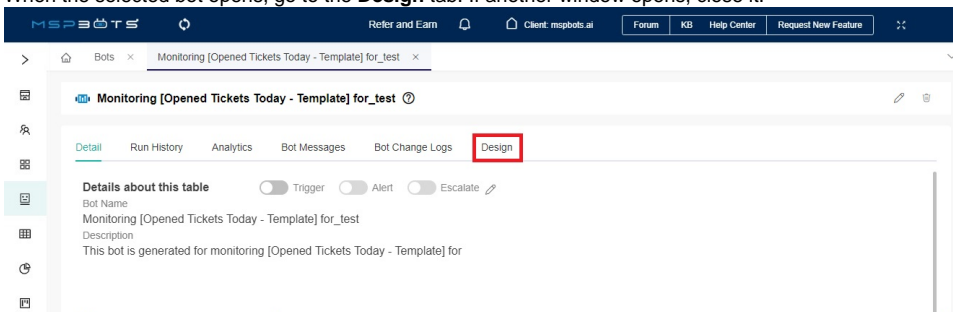
1. Go to **Bots** on the MSPbots menu.



2. Open the bot where you need to add a REST API Call block. Let's use the bot **Monitoring [Opened Tickets Today - Template] for\_test** as an example.



3. When the selected bot opens, go to the **Design** tab. If another window opens, close it.



4. In the trigger block, configure the widget or dataset that you have already created as a data source according to your needs. For detailed configuration information of the trigger block, please refer to [2. Set up the bot trigger](#).

**Trigger** ⓘ

Define the bot and the conditions for when it will execute.

I want the bot to trigger when

Dataset Teams / Office 365 AD users

meets the following criteria:

AND OR + -

user\_name Equals @mspb

I want the bot to run based on this schedule: Asia/Shanghai

Starting at 2022-11-09 03:33:03

Repeat every 3 Hour

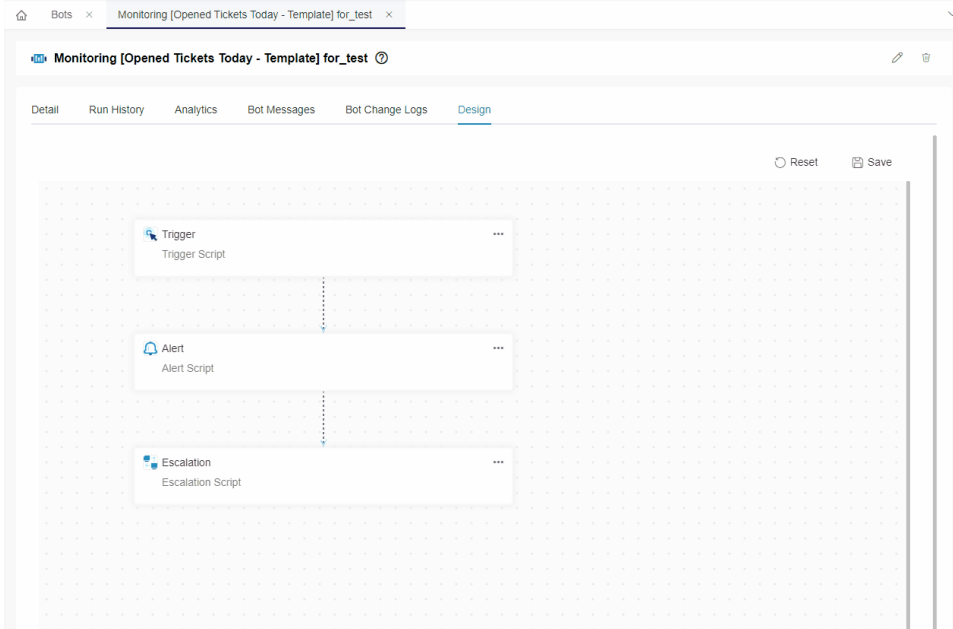
Repeat every 10 minutes

from hour 00:00 to 23:00 on day of week Monday +4

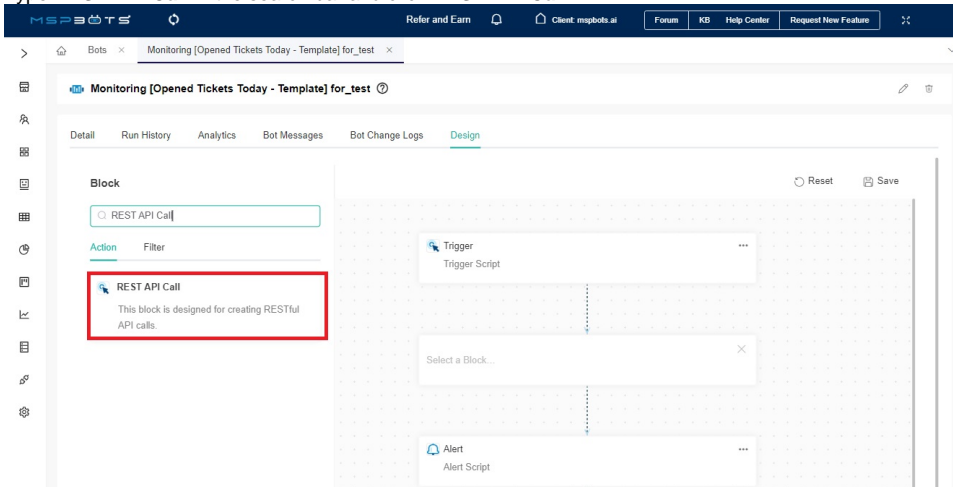
Advanced Scheduler 0 0 3/5 \* \* ? \* [Setting →](#)

[Next](#)

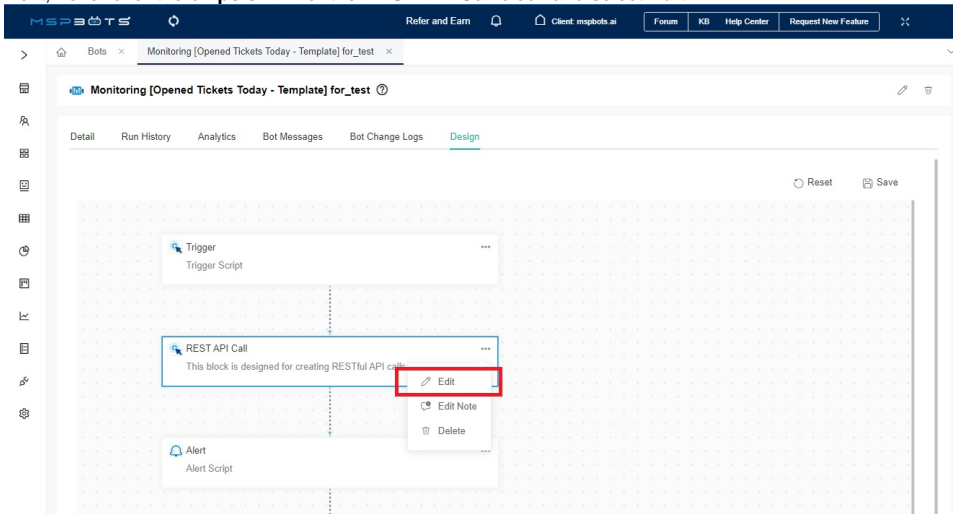
5. Hover over the horizontal line between the blocks. Click , select **Add a Block**, and select the new block.



6. Type **REST API Call** in the search bar and click **REST API Call**.




7. Next, hover over the **ellipsis** \*\*\* for the REST API Call block and select **Edit**.



8. When the REST API Call window opens, enter the **url** address of the API that will receive requests you want to call, for example, <https://mspbots.st.halopsa.com/api/Tickets>.

REST API Call



This block is designed for creating RESTful API calls.:

This block is designed for creating RESTful API calls, enabling direct interaction with RESTful APIs. Warning: This is an advanced feature, and should be used with caution. For assistance, please contact support@mspbots.ai.

uri:

integration:

method:

headers:

key	value
No Data	

params:

key	value
No Data	

query

body

key

value

Add

Add


Previous

Next

9. (Optional) Select an **integration**.

If you are configuring a REST API from an existing integration with MSPbots, select that software from the **integration** list, such as *halo*. Otherwise, do not configure this parameter and proceed with the next step.

REST API Call



This block is designed for creating RESTful API calls.:

This block is designed for creating RESTful API calls, enabling direct interaction with RESTful APIs. Warning: This is an advanced feature, and should be used with caution. For assistance, please contact support@mspbots.ai.

uri:

integration:

method:

headers:

key	value
No Data	

params:

key	value
No Data	

key

value

Add

Add


Previous

Next

10. Select the HTTP **method** used when calling the API. The commonly used methods when using this block are **post** and **delete**. If you want to learn examples of these two methods, please refer to [Examples of commonly used HTTP Methods when using the REST API Call block](#).
- **get** - Retrieves information from the database without modifying or adding data. The results are consistent regardless of how many operations are performed.
  - **post**: Submits data and adds operations to the server.
  - **put**: Modifies the existing data on the server. This is similar to POST except that it modifies instead of adding.
  - **delete**: Removes a specific resource and deletes a record in a database.

- **options:** Before using any of these methods, you can check the browser request to ensure the server will accept it.

REST API Call



This block is designed for creating RESTful API calls.:

This block is designed for creating RESTful API calls, enabling direct interaction with RESTful APIs. Warning: This is an advanced feature, and should be used with caution. For assistance, please contact support@mspbots.ai.

uri:

integration: Select

method: get

headers:

key

value

No Data

Add

params:

key

value

No Data


Add

Previous

Next

- Next, set up the **headers** and input the relevant **key** and **value**. Headers are additional information included in the API call, which may include authentication tokens and content types.  
As our Bot already supports authentication information by default, you don't need to add any extra authentication information to REST API Call block.

REST API Call



This block is designed for creating RESTful API calls.:

http client

uri:

integration: Select

method: get

headers:

key

value

Add

Delete

query

body

params:

key

value


No Data

Add

Previous

Next

Click **Add** to add more keys and values.



This block is designed for creating RESTful API calls.:

http client

url:

integration:

Select

method:

get

headers:

key	value	
<input type="text"/>	<input type="text"/>	<div>Add</div> <div>Delete</div>

query

body

params:

key	value	
No Data		

Previous


Next

To remove a pair of key and value, click **Delete**.

12. Set up the **params**. Select either the URL query and request body method to pass parameters to the server.

**a. query**

URL query parameters refer to additional parameters or information passed to the server through key-value pairs attached to the URL. Query parameters are commonly used for filtering, sorting, pagination, or passing other optional parameters. They are placed after the question mark symbol (?) in the URL and key-value pairs are separated by an equals sign (=), while multiple query parameters are connected with an ampersand (&). By utilizing URL query parameters, we can pass additional parameters in the API call and customize the results requested.



This block is designed for creating RESTful API calls.:

http client

url:

integration:

Select

method:

get

headers:

key	value	
<input type="text"/>	<input type="text"/>	<div>Add</div> <div>Delete</div>

query

body

params:

key	value	
No Data		

Previous

Next

**b. body**

The request body is the HTTP request data that is included in a POST or PUT request. It is only configured when the **method** is **POST** or **PUT**. This data can be in different formats such as JSON, XML, or form data. These parameters are usually used for submitting forms, uploading files, or sending other types of data.

This block is designed for creating RESTful API calls.:  
http client

uri:

integration: Select

method: get

headers:

key	value	
<input type="text"/>	<input type="text"/>	<span>Add</span>
		<span>Delete</span>

query

**body**

params:

☒ none
 ☐ form-data
 ☐ x-www-form-urlencoded
 ☐ json
 ☐ xml

Previous
Next

The params available for the request body method are **none**, **form-data**, **x-www-form-urlencoded**, **JSON**, and **XML**.

This block is designed for creating RESTful API calls.:  
http client

uri:

integration: Select

method: get

headers:

key	value	
<input type="text"/>	<input type="text"/>	<span>Add</span>
		<span>Delete</span>

query

**body**

params:

☒ none
 ☐ form-data
 ☐ x-www-form-urlencoded
 ☐ json
 ☐ xml

Previous
Next

- Click **Next** to configure any remaining blocks, and click **Finish** when done.
- Click **Save** to keep your settings.

## Examples of commonly used HTTP Methods when using the REST API Call block

### post - Create tickets to the Halo app with the filtered data that meets the criteria.

\*For more detailed information, please refer to: [How to Create or Modify Halo Tickets using a Rest API-based Bot](#).


Parameter Field	Description
url	Fill in the interface address for creating or modifying tickets in Halo: <b>https://{host}/api/Tickets</b> , for example: <a href="https://mspbtest.halopsa.com/api/Tickets">https://mspbtest.halopsa.com/api/Tickets</a> .
Integration	Halo
method	post
headers	Can be left empty.
params	<ol style="list-style-type: none"> <li>Click <b>body</b>.</li> <li>Select <b>json</b>.</li> </ol>

3. Input parameter fields in the JSON body, using the following columns as an example to create tickets.

```
[
  {
    "actioncode":0,
    "dateoccurred":"2023-12-15T14:35:55.618Z",
    "summary":"{job_title}",
    "details":"{user_name}",
    "status_id":"2",
    "tickettype_id":"{update_source}"
  }
]
```

- About the example:
  - The "actioncode", "dateoccurred", "summary", "details", "status\_id", "tickettype\_id" are from the official API documentation of Halo, <https://halo.haloservicedesk.com/apidoc/resources/tickets>. These fields are generally obtained from the official API documentation of the integration.
  - The {job\_title}, {user\_name}, {update\_source} are from the data you added in the widget or dataset in the Trigger block. Please note that the format of the field must be **{Field}**.

REST API Call ⓘ



http client

url:

https://mspbotstest.halopsa.com/api/Tickets

integration:

Halo

method:

post

headers:

key	value
No Data	

params:

querybody

noneform-datax-www-form-urlencodedjsonxml

```
[
  {
    "actioncode":0,
    "dateoccurred":"2023-12-15T14:35:55.618Z",
    "summary":"{job_title}",
    "details":"{user_name}",
    "status_id":"2",
    "tickettype_id":"{update_source}"
  }
]
```

Previous


Finish

delete - Delete tickets in the Halo app that meet the filtering criteria.

url	Please fill in the interface address to delete tickets in Halo: https://{host}/api/Tickets, for example: <a href="https://mspbotstest.halopsa.com/api/Tickets/{id}">https://mspbotstest.halopsa.com/api/Tickets/{id}</a>
Integr ation	Halo



method	delete
headers	Can be left empty.
params	<ol style="list-style-type: none"> <li>1. Click <b>body</b>.</li> <li>2. Select <b>json</b>.</li> <li>3. Input parameter fields in the JSON body, using the following columns as an example.</li> </ol> <pre>[   {     "id": "{ticket_id}"     "reason": "{delete_reason}"   } ]</pre> <ul style="list-style-type: none"> <li>• About the example: <ul style="list-style-type: none"> <li>◦ The "id", "reason" are from the official API documentation of Halo, <a href="https://halo.haloservicedesk.com/apidoc/resources/tickets">https://halo.haloservicedesk.com/apidoc/resources/tickets</a>. These fields are generally obtained from the official API documentation of the integration.</li> <li>◦ The {ticket_id}, {delete_reason} are from the data you added in the widget or dataset in the Trigger block. Please note that the format of the field must be <b>{Field}</b>.</li> </ul> </li> </ul>



http client

REST API Call ⓘ

×

uri:

https://mspbottest.halopsa.com/api/Tickets/{id}

integration:

Halo ▾

method:

delete ▾

headers:

key	value
No Data	

Add

query

body

none

form-data

x-www-form-urlencoded

● json

xml

params:

```
[
  {
    "id": "{ticket_id}"
    "reason": "{delete_reason}"
  }
]
```

Previous

Finish

## Related topics

- [How to Create or Modify Halo Tickets using a Rest API-based Bot](#)
- [How to Customize the Commands Used in Chat](#)