How To Use Webhooks With Slack

Brief Overview

Webhooks (AKA callbacks) allow you to trigger 3rd party APIs and/or add WalkMe Event data into your analytics platform. The webhook is triggered by a designated WalkMe Event, for example, a ShoutOut being displayed. The webhook integrates between WalkMe and your chosen 3rd party platform.

This article explains how to define webhooks for Slack.

Use Cases

Webhook for Slack use cases include the following:

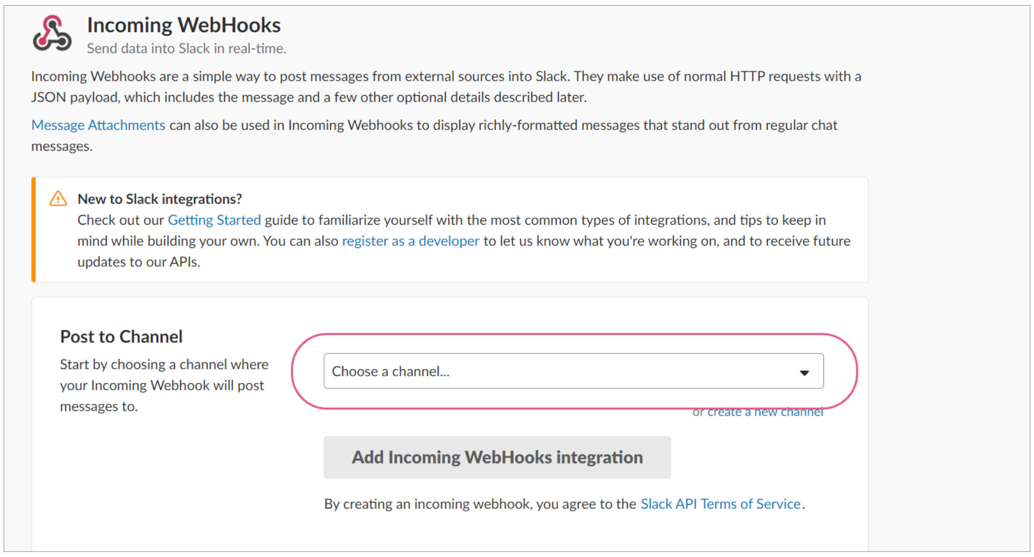
- Send any WalkMe Event (i.e., clicks on specific WalkMe elements) you wish to track to the Slack channel of your choosing;
- Receive real-time notifications when specified events occur;
- Better understand how your end-users engage with WalkMe in real-time in order to meet their needs and follow their journeys;
- Utilize the vast Slack API library to manipulate and utilize your incoming webhooks as you please.

Steps for Defining a Webhook for Slack

Step 1: Create a Slack channel

Step 2: Add a webhook to the Slack channel

1. Type "Apps" into the Slack search bar, and click on *Apps;*
2. Click *Manage apps*... on the top left;
3. In the search bar, type "Incoming Webhooks" and click *Incoming Webhooks;*
4. Click the green *Install* button;
5. Click the dropdown menu in Post to Channel and select the channel you created in Step 1;


Incoming WebHooks
Send data into Slack in real-time.

Incoming Webhooks are a simple way to post messages from external sources into Slack. They make use of normal HTTP requests with a JSON payload, which includes the message and a few other optional details described later.

Message Attachments can also be used in Incoming Webhooks to display richly-formatted messages that stand out from regular chat messages.

New to Slack integrations?
Check out our Getting Started guide to familiarize yourself with the most common types of integrations, and tips to keep in mind while building your own. You can also register as a developer to let us know what you're working on, and to receive future updates to our APIs.

Post to Channel
Start by choosing a channel where your Incoming Webhook will post messages to.

Choose a channel...
or create a new channel

Add Incoming WebHooks integration

By creating an incoming webhook, you agree to the Slack API Terms of Service.

6. Click *Add Incoming Webhooks integration*;
7. Click *Save Settings*.

Step 3: Enter into the Slack channel you created

Step 4: Click “incoming-webhook”

Step 5: Click “Settings”

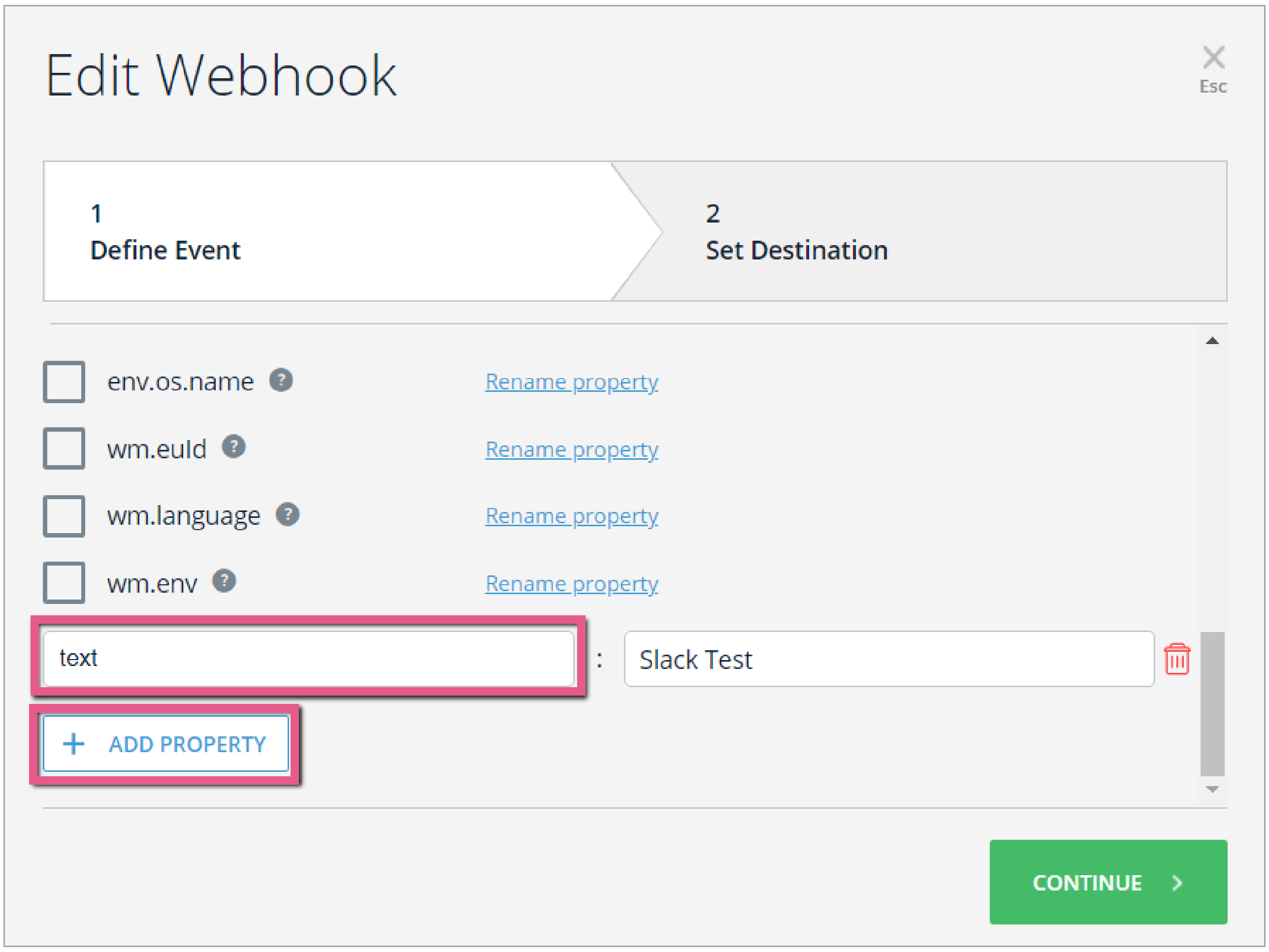
Step 6: Click “Copy URL”

Step 7: Review ‘How To Send WalkMe Event Data To 3rd Party Systems Using Webhooks‘ to learn how to define your new incoming webhook on Insights;

Step 8: Proceed using the Webhook Wizard

Wizard — Step 1: Define Event

- Events are sent to Slack by the “text” property;
- **Click “+ ADD PROPERTY” and add the “text” property;**



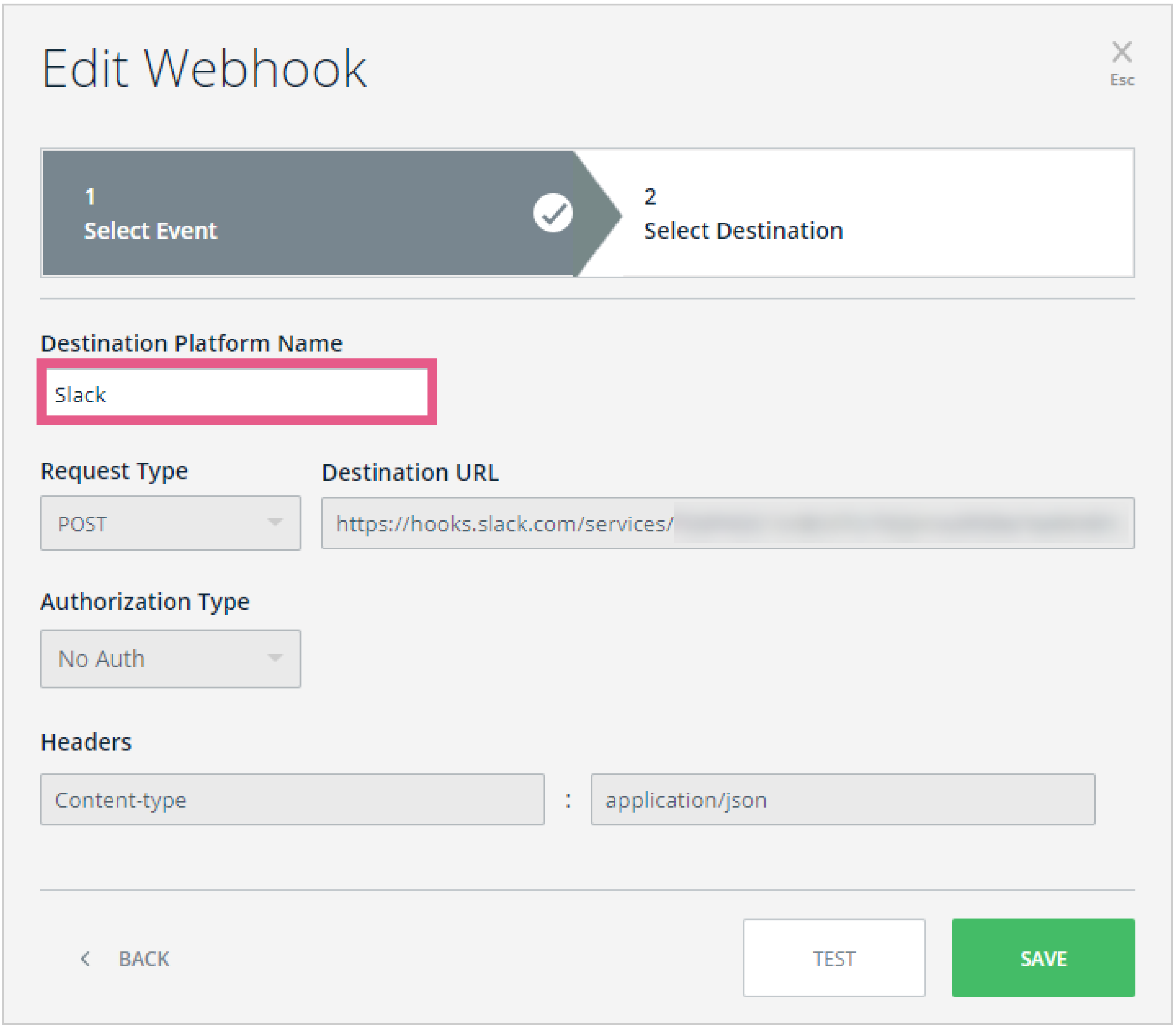
text : Slack Test

Pro-Tip

Construct the sentence you want to send to your Slack channel and incorporate data from the predefined properties by adding “@” and selecting the property name from the list that appears.

Wizard — Step 2: Set Destination

- **Destination platform Name**: This will be used to identify the destination system (in this case Slack);



Edit Webhook

1 Select Event

2 Select Destination

Destination Platform Name: Slack

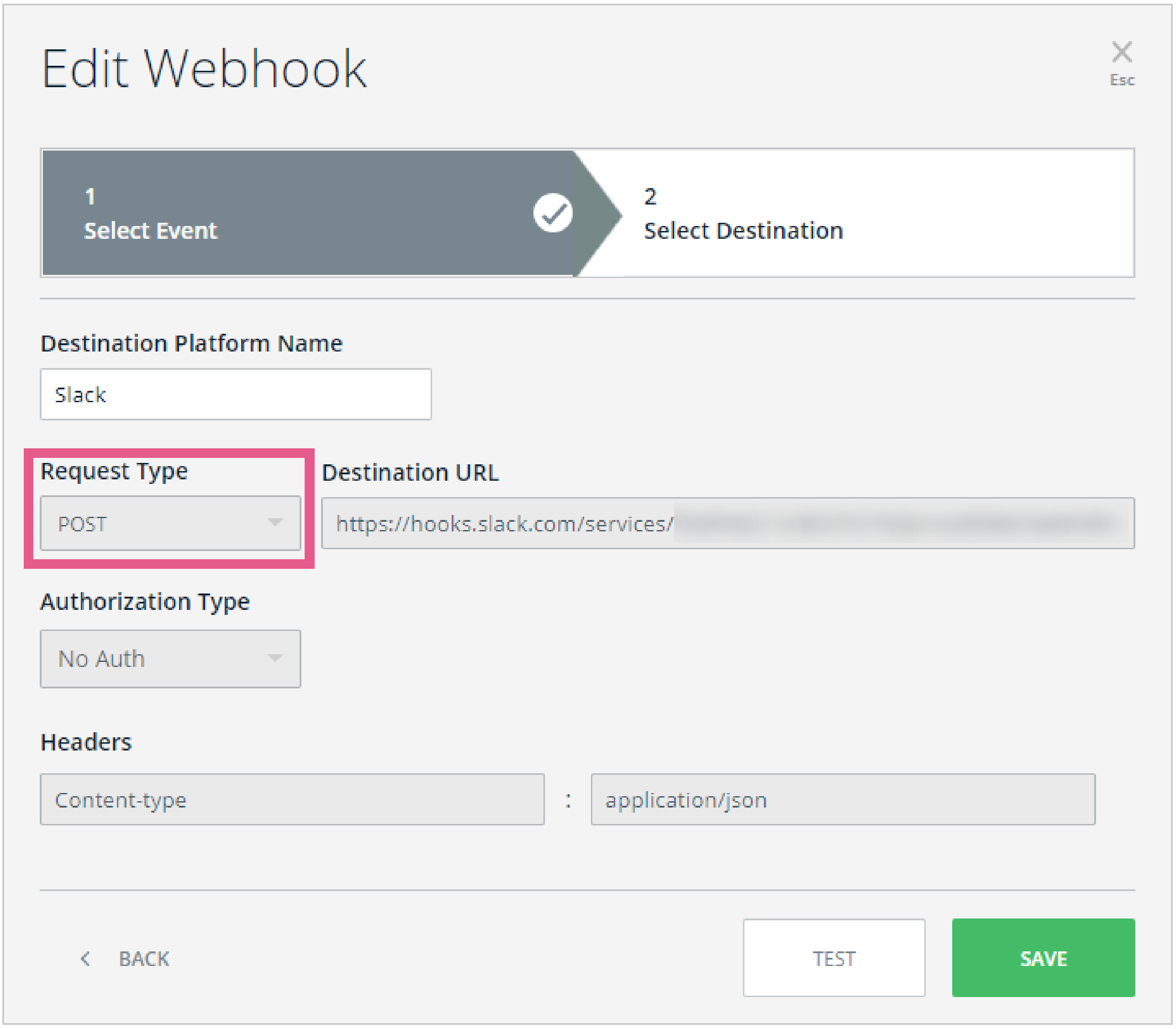
Request Type: POST

Destination URL: https://hooks.slack.com/services/[redacted]

Authorization Type: No Auth

Headers: Content-type : application/json

- Set Request Type to “POST”;


Edit Webhook

1 Select Event

2 Select Destination

Destination Platform Name: Slack

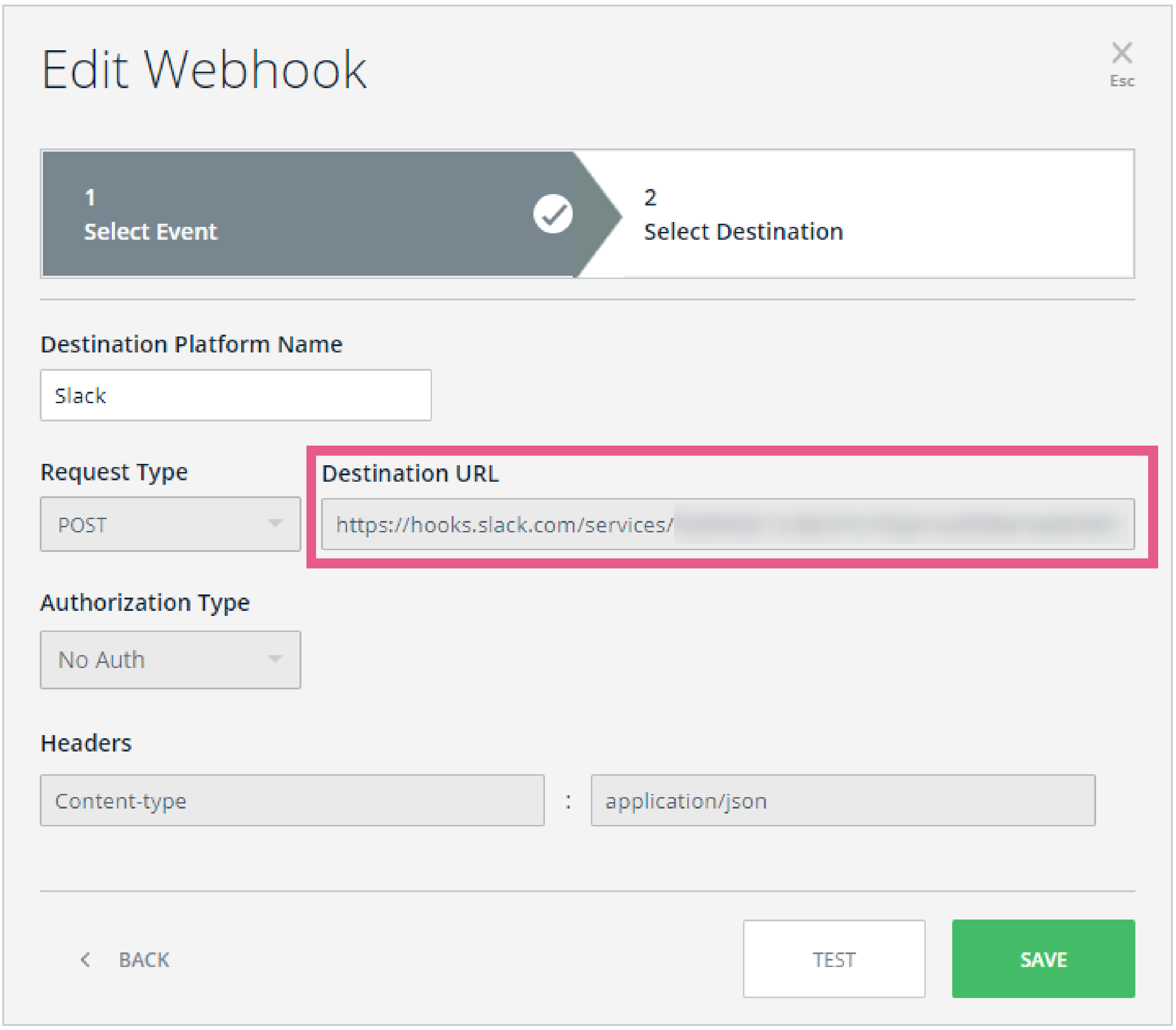
Request Type: POST

Destination URL: https://hooks.slack.com/services/

Authorization Type: No Auth

Headers: Content-type : application/json

- Paste the Destination URL you copied in **Step 6**;



Destination Platform Name: Slack

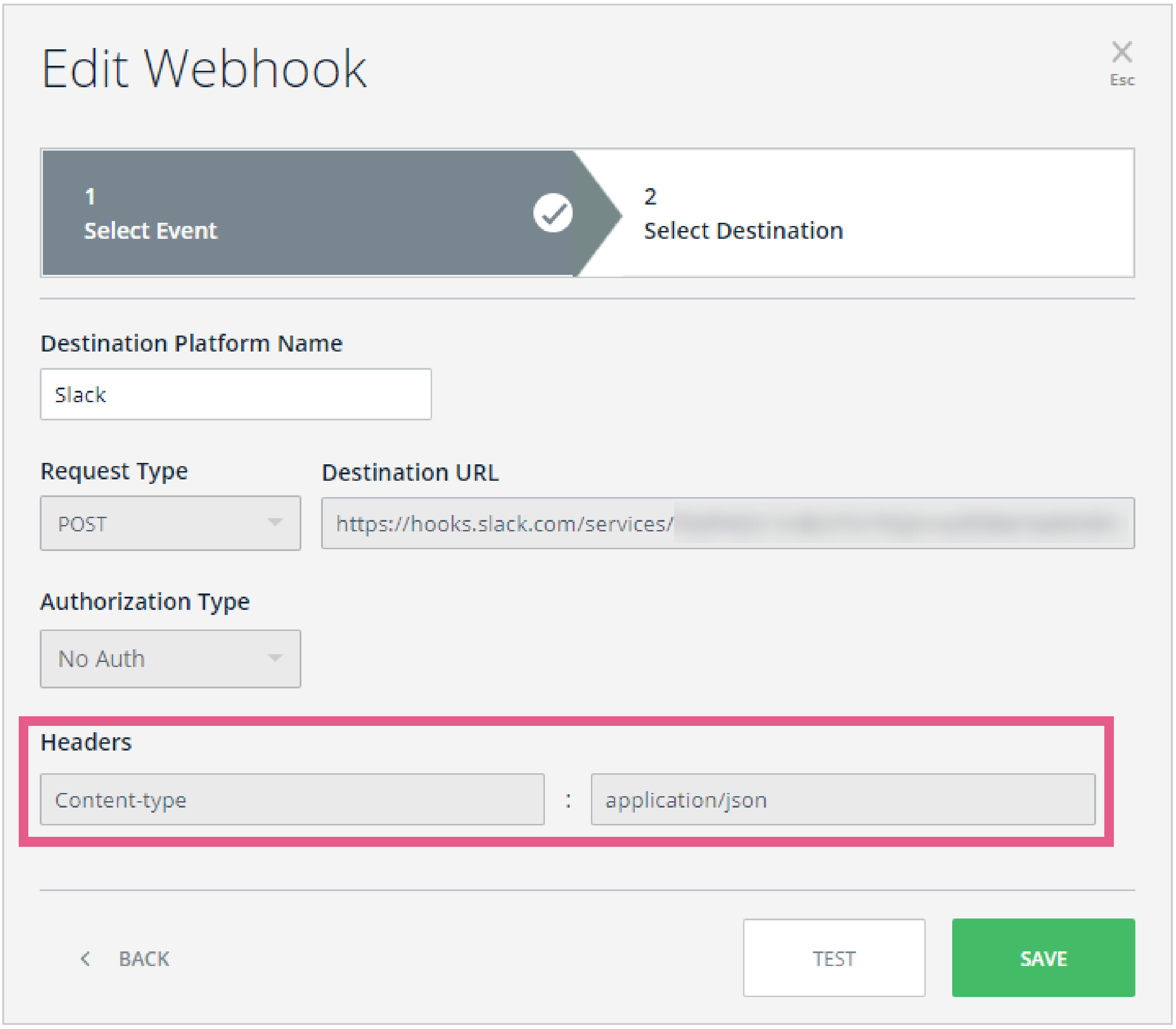
Request Type: POST

Destination URL: https://hooks.slack.com/services/[redacted]

Authorization Type: No Auth

Headers: Content-type : application/json

- Add Header – Content-type : application/json;



Edit Webhook

1 Select Event

2 Select Destination

Destination Platform Name: Slack

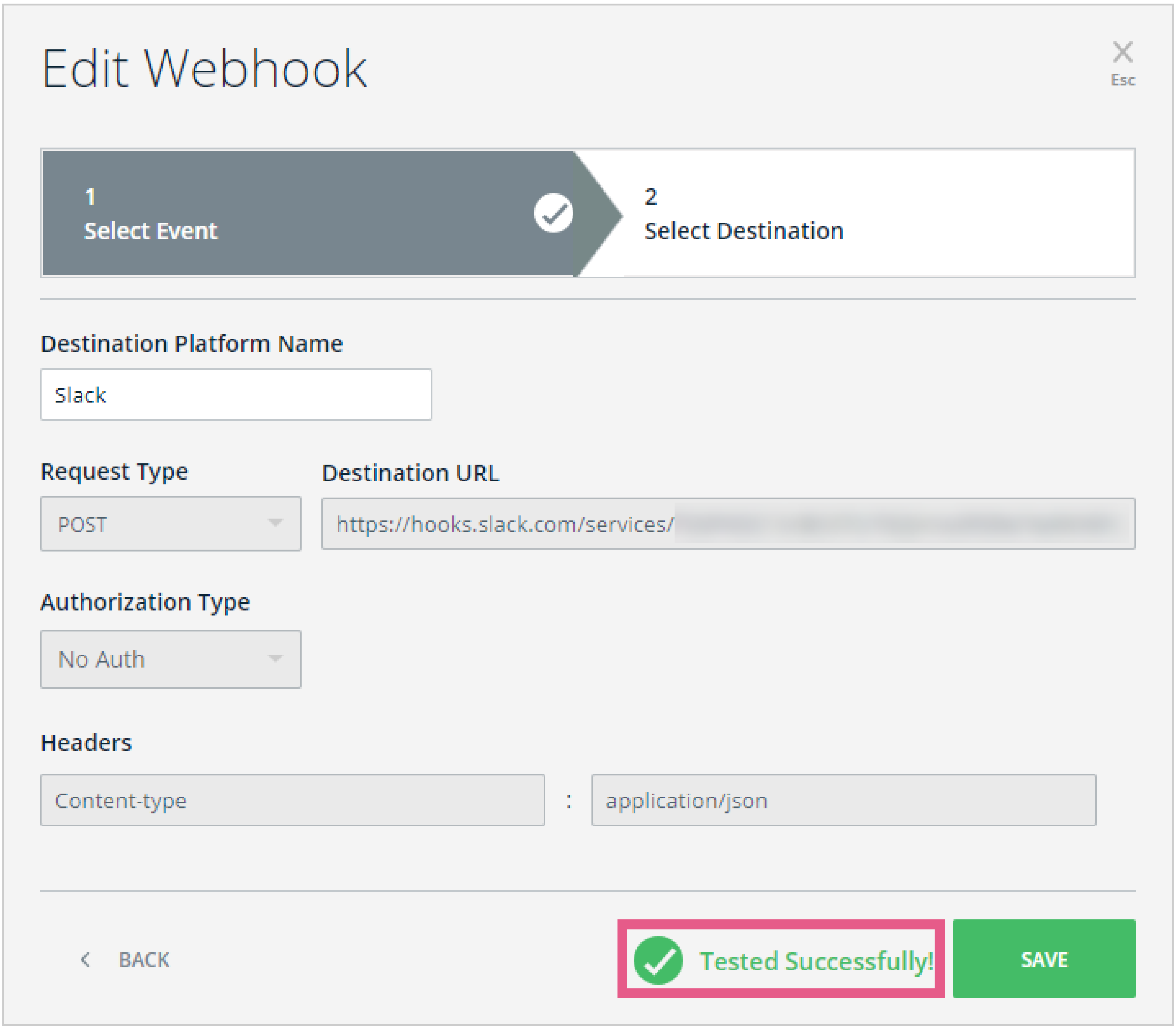
Request Type: POST

Destination URL: https://hooks.slack.com/services/[redacted]

Authorization Type: No Auth

Headers: Content-type : application/json

- Click *Test*. You should see the message “Tested Successfully!” in green;



Edit Webhook

Esc

1 Select Event ✓

2 Select Destination

Destination Platform Name: Slack

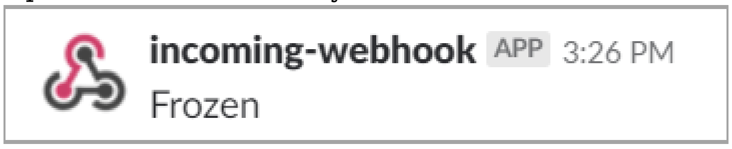
Request Type: POST

Destination URL: https://hooks.slack.com/services/[redacted]

Authorization Type: No Auth

Headers: Content-type : application/json

- Open the Slack channel you defined and see that webhooks sent a random word;



incoming-webhook APP 3:26 PM
Frozen

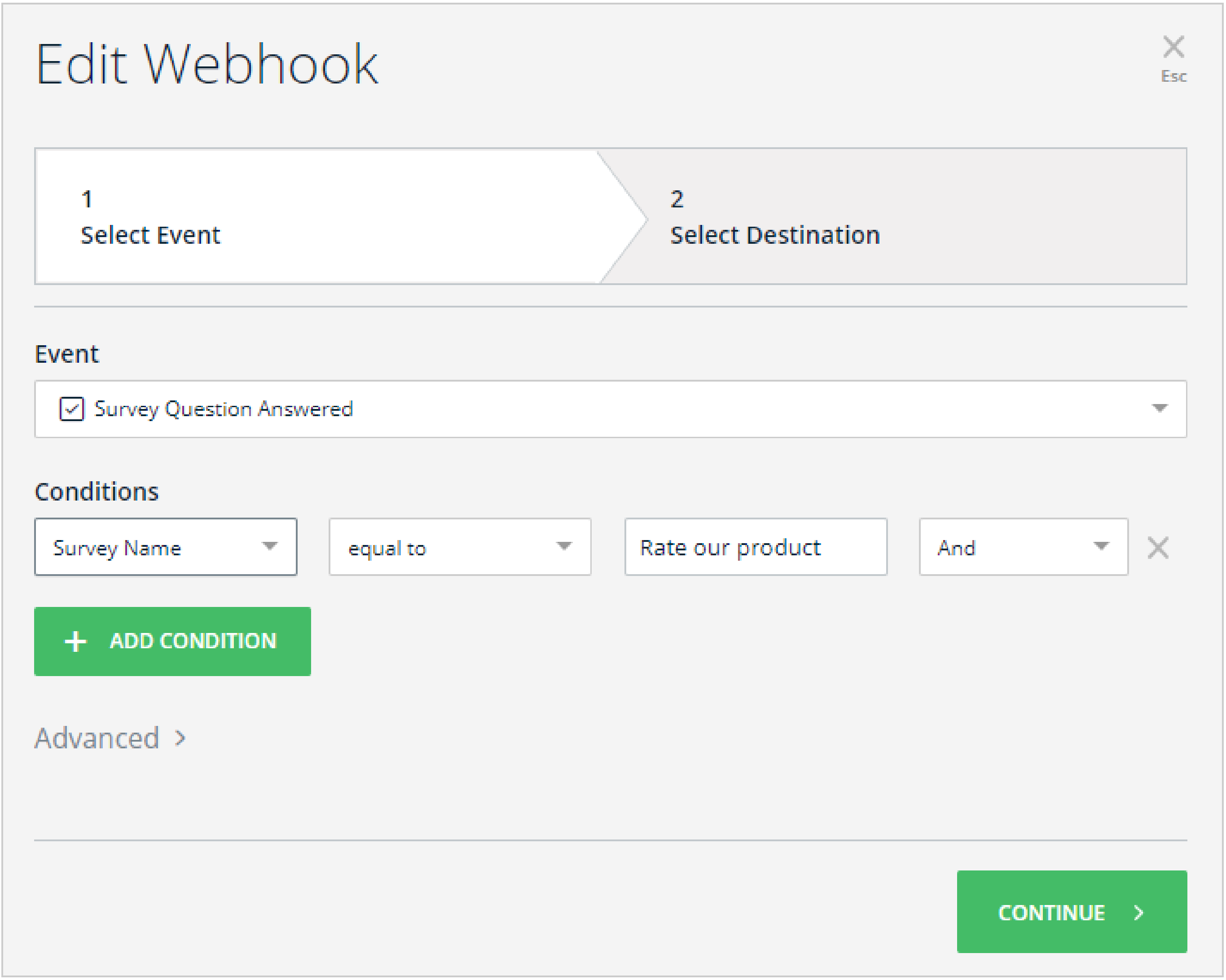
- Click *SAVE;*
- FINALLY, try running the event you defined and observe the result on Slack!

Pro-Tip

Read [this Slack Article] to make your webhook fancy using advanced formatting!

Example Implementation: Sending Survey Responses to Slack

Step 1: Choose Survey Question Answered event from Event



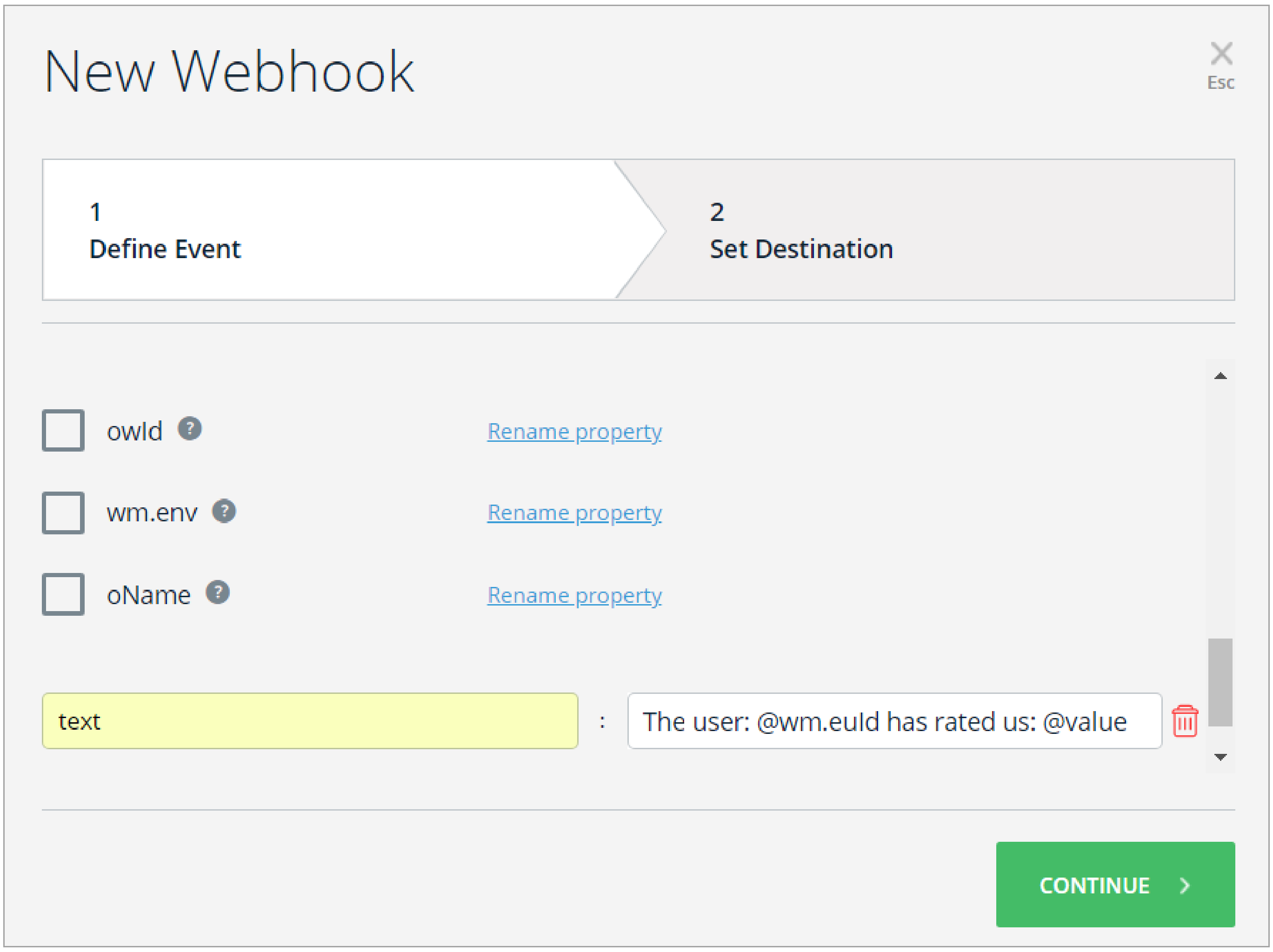
Edit Webhook

Event: Survey Question Answered

Conditions: Survey Name | equal to | Rate our product | And

Step 2: Uncheck all the properties of the event

Step 3: Add property “text” and set this as its value: The user: @wm.euId has rated us: @value



New Webhook

Esc

1 Define Event

2 Set Destination

- [ ] owId
- [ ] wm.env
- [ ] oName

Rename property

text : The user: @wm.euId has rated us: @value

CONTINUE