Lead Capture Forms

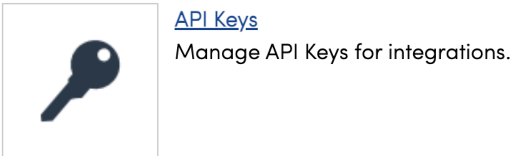
07/16/2025 1:51 pm CDT

Overview

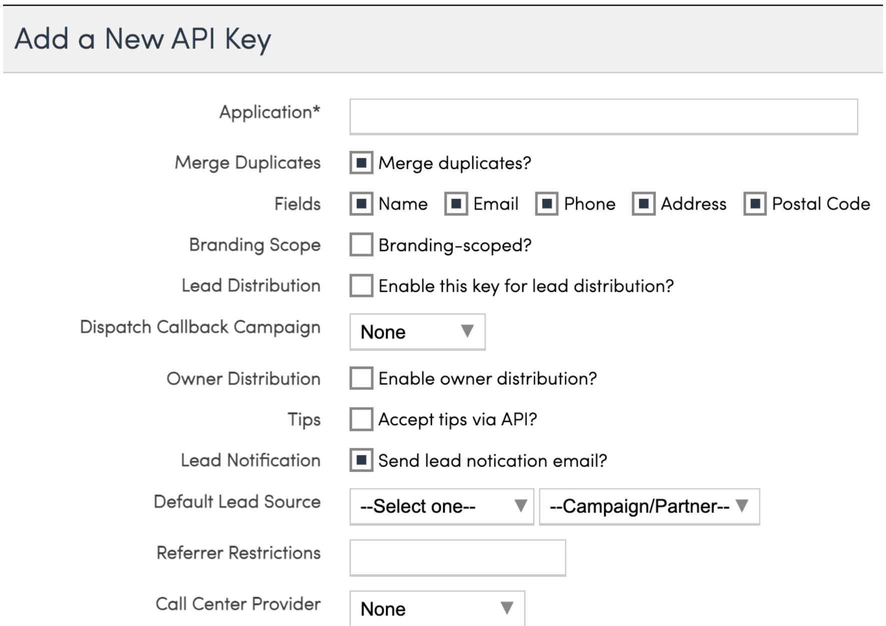
You can post data directly to service**minder**.io via HTTP FORM POSTs. This is most often used to set up lead capture forms on your website or on a digital marketing landing page. You can use this method to integrate with landing pages hosted on sites such as InstaPage and Unbounce. Just design a form and make sure the field names match the ones shown below (plus any custom contact fields you have set up). Then set the action to a URL that points to serviceminder.io.

https://serviceminder.io/service/contact/addupdate/*APIKey*

You'll first need to create an API key. You can do this by going to **Control Panel > API Keys** and clicking **Add**.



You'll provide the basic details about your API Key and configure a few specific settings for it.



For a Lead Capture form, the important fields to consider are:

- **Merge Duplicates -** If leads come in with the same details, do you want them to automatically be merged with the existing contact?
  - **Fields -** Which fields do you want to cross-reference when deciding to merge?

> If a duplicate contact is merged with a different lead source from the original contact record, this new lead source will be tracked in the system and tied to the invoice. Learn more about that HERE.

- **Lead Distribution -** This decision is made at the brand level when deciding to distribute leads to various territories/locations within the brand. If you are a single organization, do not check this box.
- **Owner Distribution -** If you have Contact Owners enabled, you can distribute leads that come in via this API key in a round-robin style.
- **Lead Notification -** If this box is checked, an email will be sent to the designated Lead Notification email address (set up in Control Panel > Notifications).
- **Default Lead Source -** This will place all leads that come in from this API key into a specific Channel and Campaign, if designated.

Once you have your API key, you'll then configure your form's **ACTION** to point to the **URL**. We recommend creating a separate API key for each form you create. This will allow you to better track which form a contact came from.

If you enable Lead Distribution for the API key, we'll use the zip/postal code from the form to distribute the new lead] to the correct organization. You can read more about Lead Distribution here.

This can also be used to send data from one website to another. For example, in Zapier, you can specify a FORM POST action for a step that is triggered when someone completes a form on your website.

We support the following form fields:

Field	Details
Email*	
Name*	or **FirstName** and **LastName** as separate fields
Company	
Address1*	
Address2	
City*	
State*	
PostalCode*	
Phone1*	
Phone1Type	
Phone2	
Phone2Type	
Notes	A single value that will be added as a note to either the new contact or an existing contact. By default, the Title for the note will be "Form"
ContactType	Maps to Contact Category - typically “Customer” or “Prospect”
Channel	A unique channel – if it doesn’t exist, we will create it or apply a default
Campaign	A unique campaign within the channel. If it doesn't exist, we'll create it. If not supplied, we will leave blank
Tags	Comma separated list of Contact Tags to add to the contact
ReferredByHash	The contact.hash value from the referring contact
RedirectURL	If this field is included, service**minder** will return a 302 redirect response to the value in the field rather than a JSON payload with the results
HashKey	The contact.hash value from the contact you want to update. This can be embedded in a hidden field in the form

In addition to the above standard fields, you may also include custom contact fields defined in your account. The field name for each custom field is shown on the custom field details page labeled "Form Field." Custom fields will all start with a "cust_" prefix.

The format of the body must be the same as if it was a regular FORM POST -- application/x-www-form-urlencoded -- this is the default encoding for FORM POST operations so you shouldn't need to do anything special.

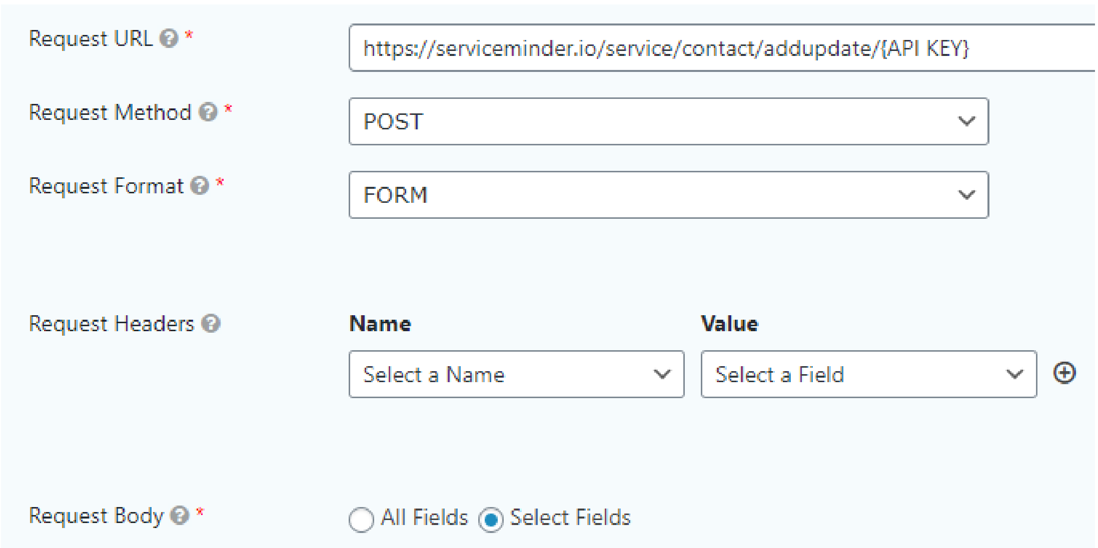
The posted form returns a JSON-formatted response with the following properties (if a RedirectUrl value is not present):

- Result => A value of 0 indicates success. A -1 indicates an unexpected error. A 1 indicates a validation error (such as missing name or email fields)
- ContactId => The internal identifier of the added (or updated) contact.
- Message => A text message with either the error info or validation info

If a value of 0 is returned for Result, then the operation was successful and you can redirect on to the next page. Any other value indicates that the data was not saved and so you should either allow the user to correct and re-submit or you may want to log the data for processing later.

Gravity Forms with Webhooks Add-on

Configuring Gravity Forms to POST to ServiceMinder is fairly straightforward. In a new Webhook Feed, place the Contact update URL with your API Key into the URL field, set the method to POST, the format to FORM, and the body to "Select Fields" so that the fields can be correctly named.



Wix Integration

If you would like to pipe a contact form on your Wix site into ServiceMinder, enter the editor and, under the "Dev Mode" menu item in the top bar, enable Corvid Dev Mode. You may now directly modify the backend and page code. You will need to create a new module in the Backend folder and include the following code:

```
import {fetch} from 'wix-fetch';

export function serviceminderPost(params_string) {
  const url = 'https://serviceminder.io/service/contact/addupdate/';
  const key = 'YOUR_API_KEY';

  let full_url = url + key;

  const headers = {
    "Content-Type": "application/x-www-form-urlencoded"
    };

  return fetch(full_url, {'method': 'post', 'headers': headers, 'body': params_string});
}
```

Then, in the code for the page containing your lead capture form, adapt the following for your form and the name of your backend form module:

```
import {serviceminderPost} from 'backend/formModule';

export function submitForm(event) {
    let FirstName = $w('#input1').value;
    let LastName = $w('#input2').value;
    let Email = $w('#input3').value;
    let Phone1 = $w('#input4').value;
    let Address1 = $w('#input5').value;
    let Address2 = $w('#input6').value;
    let City = $w('#input7').value;
    let State = $w('#input8').value;
    let PostalCode = $w('#input9').value;
    let Notes = $w('#textBox1').value;
    let params_string = `FirstName=${FirstName}&LastName=${LastName}&Email=${Email}&Phone1=${Phone1}&
Address1=${Address1}&Address2=${Address2}&City=${City}&State=${State}&PostalCode=${PostalCode}&Notes
=${Notes}`;
    serviceminderPost(params_string).then(response => console.log(response));
}
```

Once all of the correct code is in place, you'll need to hook the submit button to the submitForm function. Enable the Properties Panel underneath "Tools" in the top bar. Then, click the submit button of your form in the editor to bring up its properties. Click the plus next to the "onClick" event in the list and type the name of the submit function (e.g. "submitForm").

Your Wix form should now post straight into your ServiceMinder API key. If you are having issues with integrating your form, see the following articles for reference: https://support.wix.com/en/article/corvid-tutorial-adding-custom-interactivity-with-events

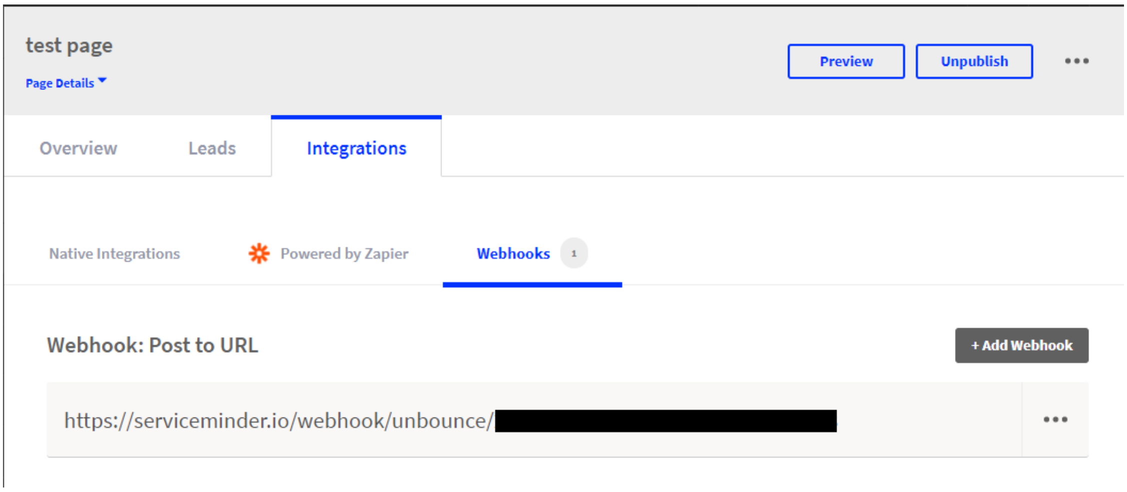
https://support.wix.com/en/article/corvid-tutorial-sending-an-email-on-form-submission

https://support.wix.com/en/article/corvid-accessing-third-party-services-with-the-fetch-api

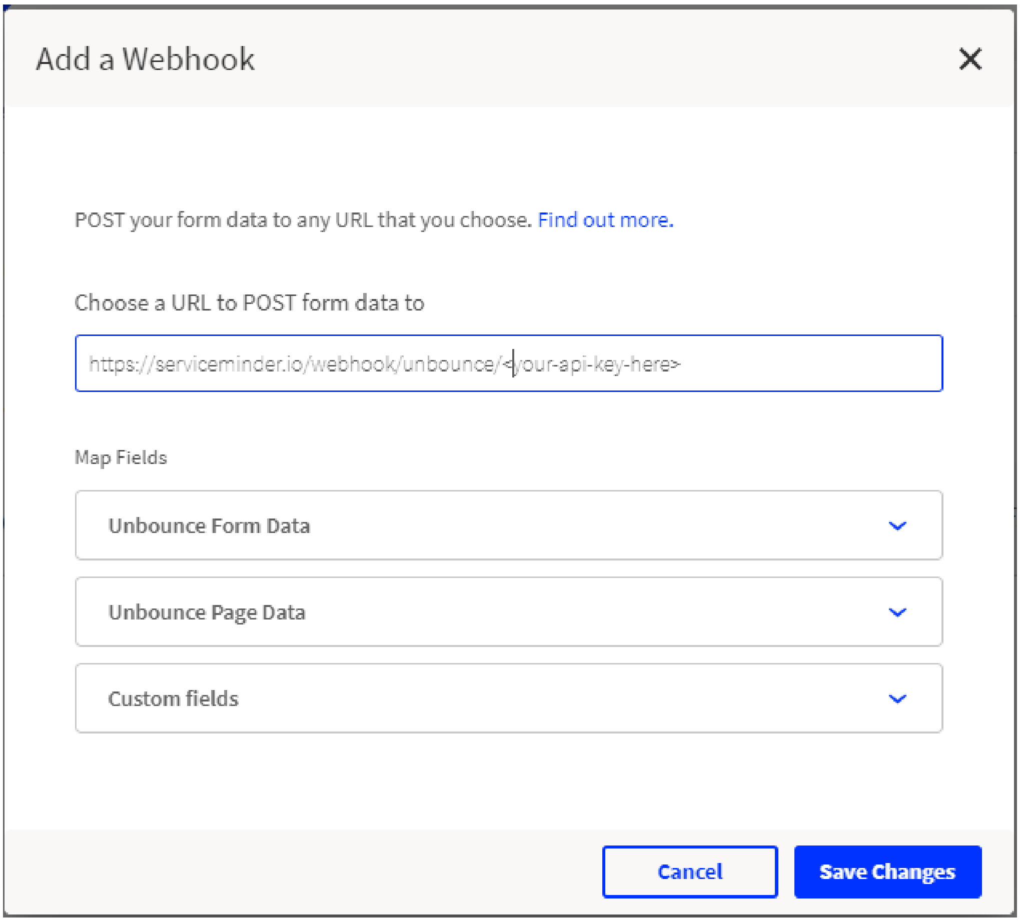
---

Unbounce Integration

service**minder**.io supports Unbounce's proprietary webhook mechanism. To connect a page hosted by Unbounce to service**minder**.io, go to the page and then select the Integrations tab.



Select **Add Webhook** and fill in the URL to point to service**minder**.io's unbounce endpoint, being sure to put your API key on the end:



Unbounce will do a quick test to make sure the URL is valid and then you're set. You can now test your form. Keep in mind that Unbounce de-dups form submissions so make sure you enter unique values for each test.

Zapier Integration

service**minder**.io supports Zapier's proprietary webhook mechanism. To send data in a Zap to service**minder**.io, you'll need an API key from service**minder**.io and then you'll need to map the Zap's fields to the above list of available fields. You can see a walkthrough of the steps for this on this page.

Google Lead Forms

To link up with Google Lead Forms, you can use a direct webhook serviceminder set up: https://serviceminder.io/webhook/googlelead/{api-key}

UTM Matching

There are actually 3 layers you can apply lead source from an API.

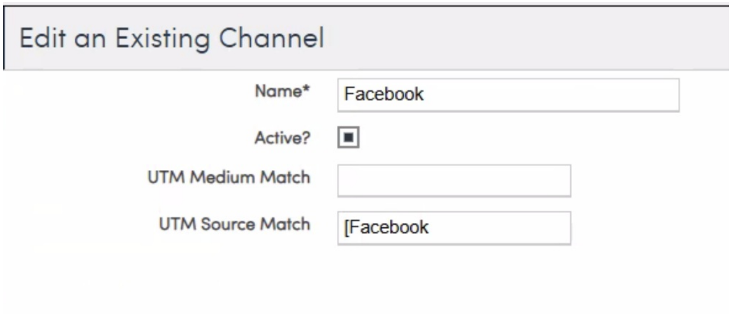
1. In the Payload
2. The API Key
3. The General Default

Payload

If passing through the API with the payload containing the source and medium, you will need to build those same channels or campaigns into ServiceMinder. Go to **Control Panel > Channels**. Then add or Edit an existing.

When adding/editing Channels or Campaigns, you can specify the UTM Match settings for:

- **utm_campaign = Medium**
- **utm_source = Channel**



The syntax for UTM parameter matching is a list of substrings separated by commas. If the UTM parameter contains any of the substrings, that will be match. You can also specify a "[" at the beginning of a substring to indicate "starts with" to allow multiple UTM parameters to match a single substring.

If multiple campaigns match, we will automatically select one (non-deterministically). If you set the UTM Medium Match, that will override the UTM Source Match.

The API Key

In **Control Panel > API**, you can set the Default Lead Source if your payload does not specify the lead source. You can set either the channel or the campaign (or both).

General Settings

In **Control Panel > General**, you can set the baseline default. Meaning, if the payload and the API key don't have a lead source set, this is the a way to ensure that the lead's attribution. You can require a Campaign or choose to leave it off.