



Whisk Studio Integration Guide

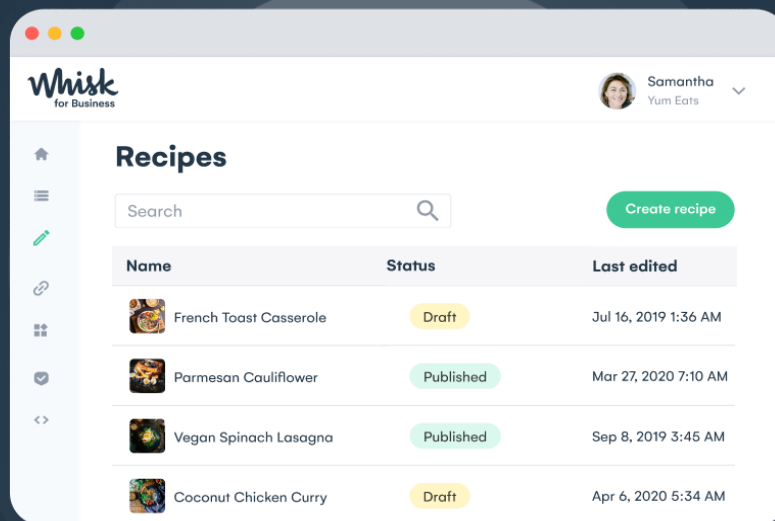


Table of Contents

Browse through our list of topics that'll help you understand how to integrate with Whisk Studio.

What is Whisk Studio?	5
Summary	5
Getting Started	6
Create an Account	6
Obtain an API key	8
Request an API key for Whisk Studio Integration	9
Explore Content Tools	10
Integration Process and Flow	13
Recipes Synchronization	14
Using the Whisk Studio App on Shopify	15
Installing the App	15
Synchronize the Recipes	15
Using the Whisk Recipes for WordPress Plugin	19
Installing the Plugin	19
Using the Whisk Studio Sync API	22
How to integrate with your publishing system?	22
Products Synchronization	23
Using Salsify Integration	24
Using the Whisk Studio App on Shopify	28
Using the Whisk Studio Sync API	28
How to integrate with your product database?	28
Content Synchronization using the Whisk Studio Sync API	30
How to set up a Custom Integration with Whisk Studio?	30
Products Synchronization	31
Recipes Synchronization	32
Recipes Sync API	32
Recipe Object	33



Core Data	34
Ingredients	35
Images	36
Source	37
Durations	38
Labels	38
Author	40
Custom Labels	41
Optional Data	42
Normalized Ingredients	42
Instructions	44
Nutrition	46
Total	47
Health Score	48
Glycemic Index	48
Glycemic Load	48
Deleted Recipe Object	49
Endpoints	50
Get Recipe	50
Query Parameters	50
Sample Curl Request	51
Sample Response	51
Get Recipe Batch	53
Query Parameters	53
Sample Curl Request	54
Sample Response	55
Products Sync API	56
Product Object	56
Core Data	58
Images	58
Responsive	59
Price	59
Recommended Retail Price (rrp)	60
Instructions	60
Ingredients	61



Author	61
Amount	62
Servings	63
Optional Data	63
Nutrition	63
Endpoints	64
Get Product	64
Query Parameters	64
Sample Curl Request	65
Sample Response	65
Get Product Batch	67
Query Parameters	67
Sample Curl Request	68
Sample Response	68
Upsert Product Batch	69
Body Parameter	70
Body Object Example	70
Sample Curl Request	72
Sample Response	72
Delete Product Batch	73
Body Parameter	73
Sample Curl Request	74
Sample Response	74
Integrations API	75
Integrations Object	75
Endpoints	76
Put Integration	77
Sample Curl Request	77
Sample Response	77
Get Integrations	78
Sample Curl Request	78
Sample Response	78
Update Integration	79
Body Parameters	80
Sample Curl Request	80



Sample Response	80
Whisk's Partners API Integration	81
Supported Integration Methods	82
You can integrate with Whisk's Partners API on either server-side or on client-side based on your requirements.	82
Server-Side Integration	82
Client-Side Integration	83
Integration Process	83
Partners API Integration Sample	84
Working with Whisk API	85
API Authentication	85
OAuth Key	85
Server Token	85
Client ID	85
Client Secret	85
Sandbox API Token	86
API Querying	86
API Documentation	86
Error Handling	86



What is Whisk Studio?

Designed for powering content strategy for leading food brands and grocers, Whisk Studio offers you a centralized platform to build and manage recipe content that's well structured, visual, consistent, and shoppable. It has tools to create, enhance, personalize and distribute recipe content, and connect it to grocery carts. It allows creators, businesses and publishers to create a unified recipe database, tied to our branded Open Food Platform. Every recipe is inspected and analyzed, allowing you to get nutrition data, truly shoppable ingredients and so much more. With Whisk Studio you can have all your recipes in one place and choose, where and how you want to publish them.

Using Whisk Studio:

- You can **manage your recipe content** by importing data from your existing database or creating new recipe content using our predefined template. It also lets you search your database quickly and easily.
- You can **optimize your recipe content** by structuring and tagging your recipes with labels such as meal type, cooking technique, cuisine, diet restrictions, and allergy restrictions. Whisk does it automatically for you and also lets you create your own custom labels. Additionally, It also provides you with automatic recipe nutrition calculation and health score using the reliable FDA data and an option to edit for maximum flexibility.
- You can **build an inspiring content experience** for your customers by accessing your content through our apps, plugins, or API and publishing it on any experience.
- You can pull all your branded and private label products into the platform or create new products using our predefined template, and **use branded products** as ingredients in your recipes.

Summary

This document describes the Whisk Studio integration process, types, and best practices.



Getting Started

Before we dive into the integration process, let's first sign-up for Whisk Studio and acquire an API key for authentication.

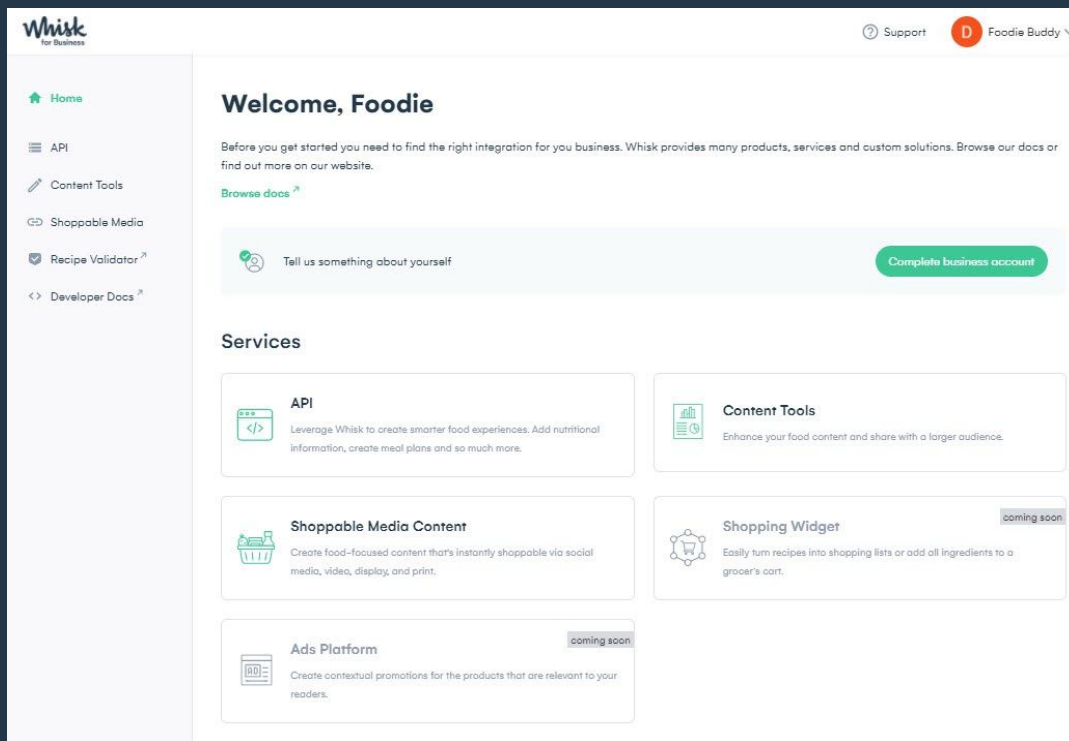
Create an Account

Whisk Studio allows anyone to create a new account.

1. Go to studio.whisk.com and sign-up using your email account.



Once registered, you get navigated to the Whisk for Business Dashboard.




2. Click the **Complete Business Account** button and finish setting up your business account profile.

The screenshot shows the 'Complete your business account' form. The top navigation bar includes 'Support' and 'Foodie Buddy'. The left sidebar is the same as the previous screenshot. The main content area is titled 'Complete your business account' and features a progress bar with three steps: 1. Profile Setup, 2. Company address, and 3. Business details. The 'Profile Setup' step is active, and the form is titled 'Enter your profile details'. It includes input fields for 'First name' (filled with 'Foodie') and 'Last name' (filled with 'Buddy'). Below these fields is a checkbox labeled 'I agree to receive marketing information at sydanish.zaidi@gmail.com', which is checked. At the bottom are 'Cancel' and 'Continue' buttons.



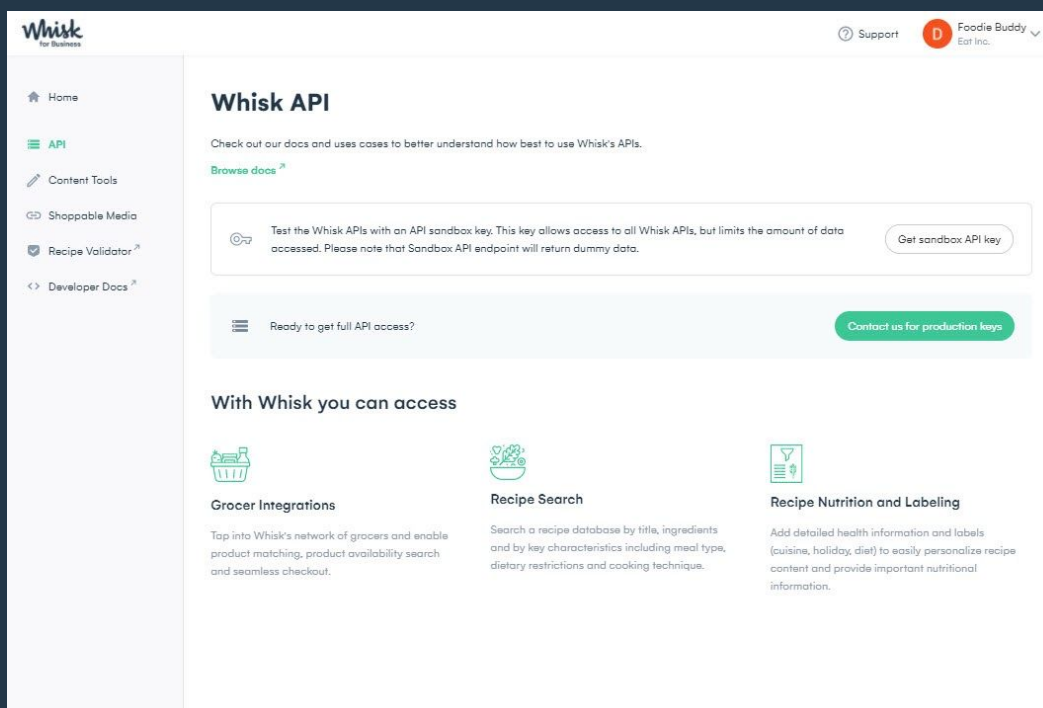
Obtain an API key

Whisk Studio provides a Sandbox key to test the Whisk Partners API. This key allows you to access all Whisk Partners API but retrieves only limited dummy data. For more information, see [Sandbox key](#).

 **Note:** The Sandbox API key is only used to test the Whisk Partners API. To authenticate your Whisk Studio Integration or Whisk Studio Synchronization API, you should obtain a production API key. See, [Request an API key for Whisk Studio Integration](#) for more information.


To obtain a Sandbox key:

1. In Whisk Studio, go to the **API** page and click **Get sandbox API key**.



2. To finish generating the API key, you must agree to the license agreement. Your sandbox key should now appear on the dashboard, and you can use it to test the Whisk API. By default, the credentials are hidden, and it's important not to share these publicly.



If you need to reset your sandbox key or have any other issues, you can contact us using the  **Support** button at the top of the page.

Finally, when you are ready to get full API access, you can click the **Contact us for production keys** button and send us your request. Someone from our team will soon reach out to you.

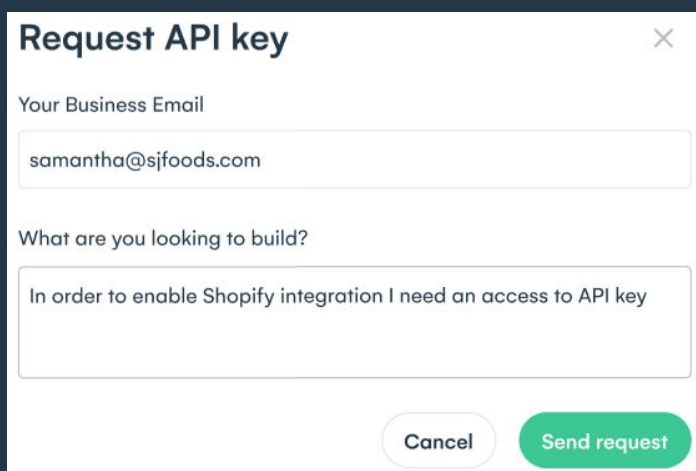
For more information on different authentication mechanisms our API support, see [API Authentication](#).

Request an API key for Whisk Studio Integration

Suppose you don't want to be involved with testing Partners API or authentication token management and only want to obtain an API key to authenticate your Whisk Studio integration. In that case, you can always request an API key from our support team.

To send us a request:

1. Log in to Whisk Studio and go to the **Integrations** page.
2. Click the integration you want to authenticate.
3. At the bottom of the page, click **Request API key**.
4. Please send us your request.



The screenshot shows a modal window titled "Request API key" with a close button (X) in the top right corner. Inside the modal, there are two text input fields. The first field is labeled "Your Business Email" and contains the text "samantha@sjfoods.com". The second field is labeled "What are you looking to build?" and contains the text "In order to enable Shopify integration I need an access to API key". At the bottom of the modal, there are two buttons: a "Cancel" button and a green "Send request" button.

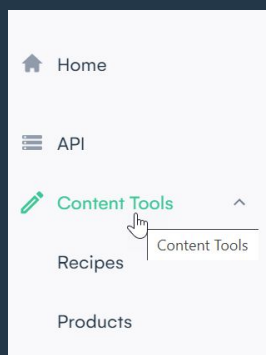
Someone from our team will soon reach out to you.


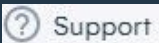


Explore Content Tools

Let's take a look at the interfaces you'll use to manage the recipes and products once you'll have the integration with Whisk Studio enabled.

In Whisk Studio, you can access the **Content Tools** from the left navigation panel.



 **Note:** If you signed up for Whisk Studio without an invite from us, you might not be able to access the Content Tools. To get access, send us the request using the  button at the top of the page and our Sales team will soon reach out to you.



Here is how the Recipes management interface looks:

Name	Status	Last edited	Created
디즈 크로켓	Draft	Dec 17, 2020 1:17 PM	Dec 17, 2020 1:17 PM
Cupcakes de banana con cobertura de limón	Draft	Dec 11, 2020 10:21 PM	Dec 11, 2020 10:21 PM
Creamy Sausage, Potato and Kale Soup	Published	Dec 11, 2020 6:00 AM	Nov 19, 2020 6:04 AM
Classic Burger	Published	Dec 9, 2020 9:54 PM	Oct 1, 2020 6:43 PM
Slow Cooker Beef Bourguignon	Draft	Dec 9, 2020 3:27 AM	Dec 9, 2020 3:27 AM
Deluxe Low Sodium Roast Beef Sandwich	Draft	Dec 9, 2020 3:19 AM	Nov 19, 2020 6:23 AM
Pudding-Chocolate Morsel Cookies	Draft	Dec 5, 2020 8:16 AM	Dec 5, 2020 1:43 AM
Copy Pudding-Chocolate Morsel Cookies	Draft	Dec 5, 2020 8:16 AM	Dec 5, 2020 1:45 AM
Prawn Cocktail Platter	Draft	Dec 4, 2020 9:28 PM	Dec 4, 2020 5:49 AM

Here is how the Products management interface looks:

Name	Barcode	Added	Last edited
Boar's Head Naturally Smoked Bacon		Dec 1, 2020	Dec 19, 2020 12:41 AM
Bob's Red Mill Gluten Free Pizza Crust		Nov 4, 2020	Nov 26, 2020 7:44 PM
Califia Farms Holiday Nog		Nov 19, 2020	Nov 19, 2020 6:13 AM
Countdown Butter Salted		Nov 13, 2020	Nov 13, 2020 5:51 AM
Fisher Pecan Halves		Nov 13, 2020	Dec 2, 2020 4:31 AM
Giant Young Turkey		Nov 13, 2020	Nov 13, 2020 12:50 AM
HEINZ Tomato Ketchup	00013000004664	Oct 1, 2020	Oct 1, 2020 6:48 PM
KRAFT SINGLE		Oct 1, 2020	Oct 1, 2020 6:40 PM
Kroger® Extra Virgin Olive Oil	00001111087864	Nov 17, 2020	Nov 17, 2020 2:45 AM
MOZZARELLA STRING CHEESE		Oct 1, 2020	Dec 7, 2020 11:10 PM

You can learn more about these tools from the Whisk Studio's documentation.



Integration Process and Flow

Whisk Studio provides plugins and API for synchronizing recipes and products.



Whisk Studio Integration Flow

For recipes' synchronization, we offer one-way sync, which means that you can use our Shopify app or WordPress plugin, or build a custom connector to pull and store recipes' metadata in your database from Whisk Studio. You can then use your preferred way to publish the recipes on your front end.

We also offer one-way sync for product synchronization, which means that you can set up an integration using Salsify or the Shopify app to push products into Whisk Studio and start using them as ingredients in your recipes.

Additionally, Whisk also provides Partners API access to its Whisk Studio users. So, you can use the Partners API along with synchronization API to build a new website or a



mobile application related to smart and shoppable recipe content.

Recipes Synchronization

Here is the general flow of how you can use Whisk Studio to manage your recipe content and publish it on your website or mobile/web application:



Recipes Synchronization Flow

There are multiple ways to integrate with Whisk Studio to synchronize your recipes:

- You can use [Whisk Recipes for WordPress](#) plugin.
- You can use [Whisk Studio app on Shopify](#).
- You can build a connector and use the Recipes Sync API to integrate with [your own publishing system](#).



Using the Whisk Studio App on Shopify

The Whisk Studio app on Shopify allows you to manage and enrich your recipes in Whisk Studio and then sync all recipes to Shopify for publishing. Similar to other apps in the Shopify marketplace, it is user-friendly and does not require any technical experience. If you are an existing Shopify merchant, you can download and install the app from the Shopify marketplace and start using Whisk Studio for managing your recipe content.

Installing the App

The Whisk Studio app is available for download in the Shopify app store for our enterprise partners.

To install the app:

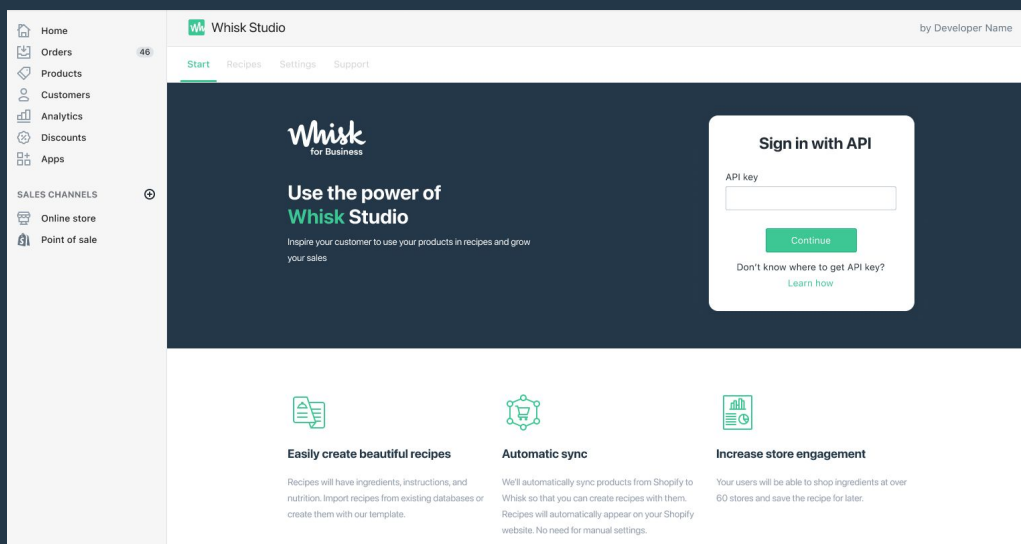
1. Go to the Shopify [app store](#) and log in to your Shopify account.
2. Search for the **Whisk Studio** app.
3. Install the app.

Synchronize the Recipes

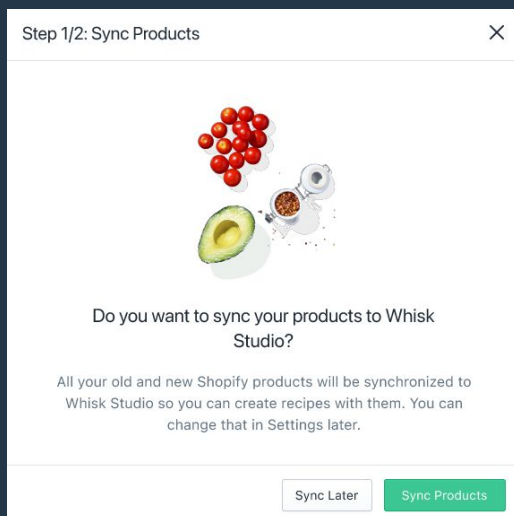
To set up synchronization:

1. On the Whisk Studio app page, authenticate using the API key to start the synchronization process. See [Obtain an API key](#) for more information.



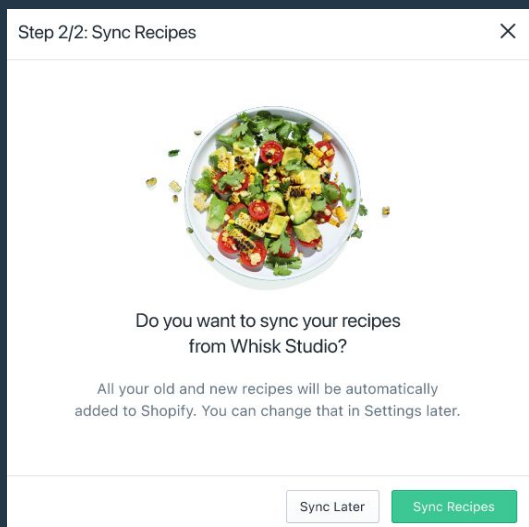


2. In the synchronization wizard that appears, click **Sync Products** to enable products' synchronization. Else, click **Sync Later**. For more information, see [Using Shopify for product synchronization](#).



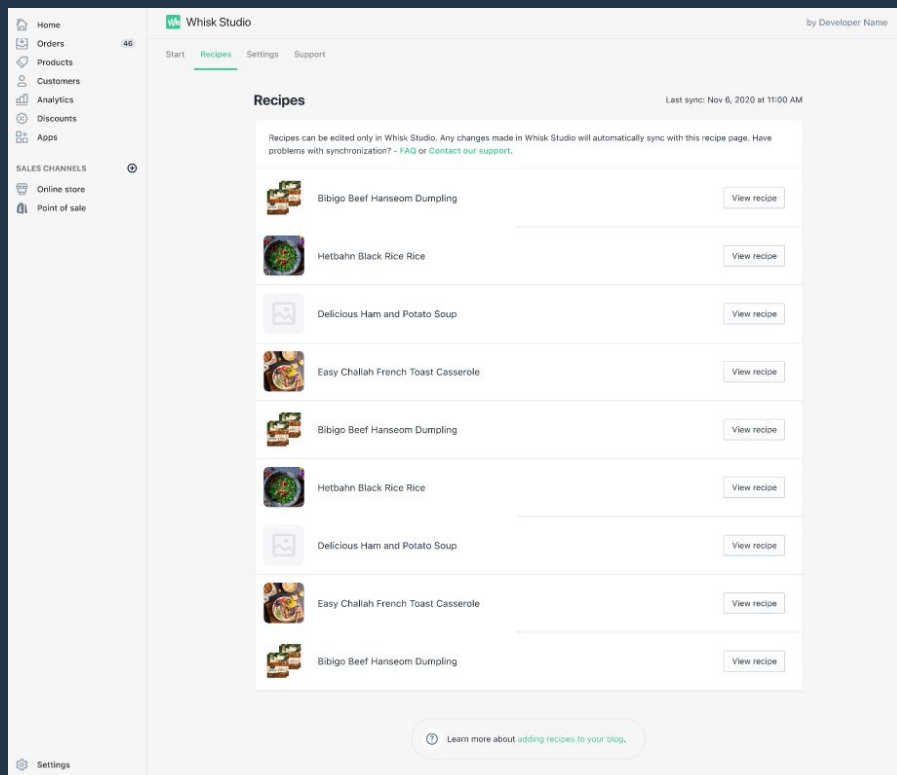
3. To synchronize recipes, click **Sync Recipes**.





All your recipe content is automatically fetched from Whisk Studio and becomes available on Shopify. If you want to skip the synchronization for now and do it later, click **Sync Later**.


Once the synchronization completes, check the **Recipes** tab on the Whisk Studio app page to verify if all the recipes are synchronized successfully.



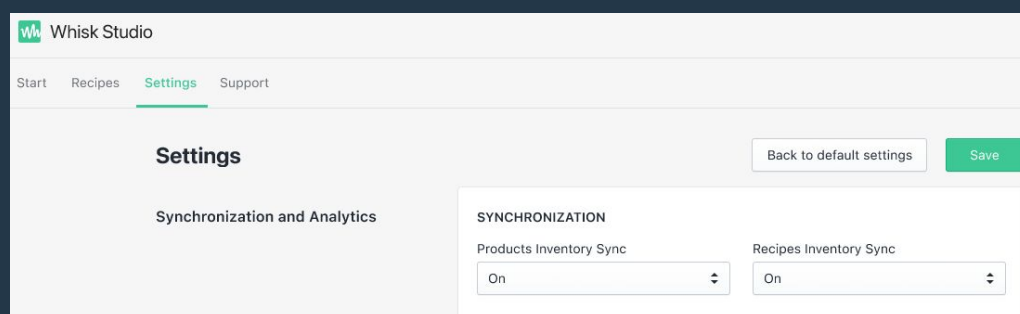
If you face any problems with synchronization, please contact our [support](#).

The Whisk Studio app stores each recipe into Shopify as an individual recipe card, which contains the following information:

- Recipe's name
- Recipe's images
- Author's name (if available)
- Recipe's description
- Ingredients list
 - Inclusive of products set as ingredients with links to take you to the products page on Shopify directly.
- Cooking instructions
 - Inclusive of images (if any).
- Servings details
- Nutrition details
- Health Score

 **Note:** You can only view the recipes in the Whisk Studio app. To edit the recipes, use the Whisk Studio platform. Any changes done in the recipe content are synchronized automatically in the Whisk Studio app on Shopify.

Irrespective of whether you synced the products during installation or not, you can still turn the synchronization on or off from the app's Settings page on Shopify.



Using the Whisk Recipes for WordPress Plugin

The Whisk Recipes for WordPress plugin automatically synchronizes the recipes you create in Whisk Studio and make them available in your WordPress account or publish them on a WordPress page based on your configuration. It also lets you change the font, sizes, and color scheme of the recipe template at any time.

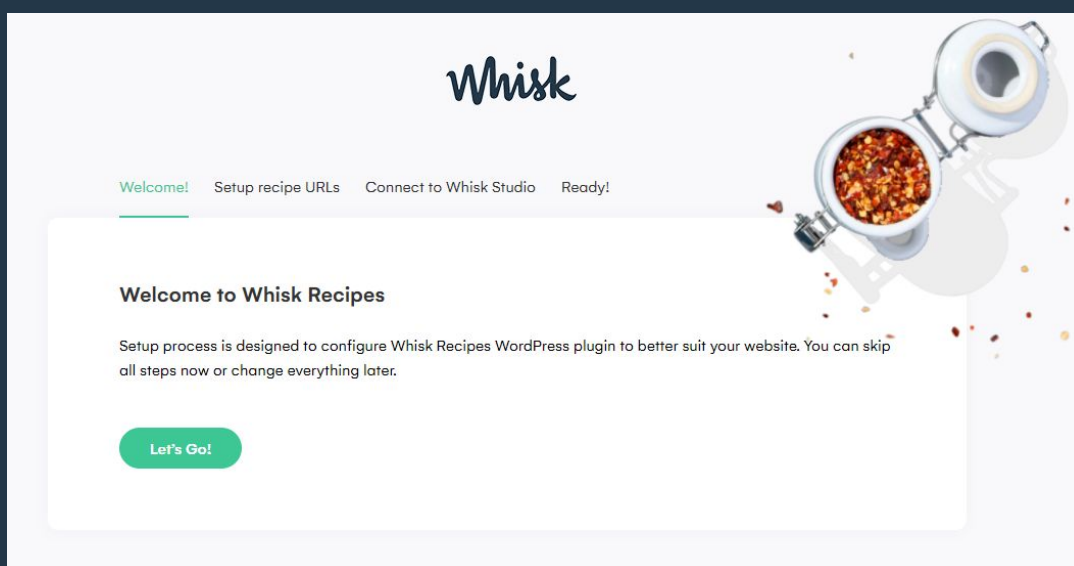
Besides being a medium to integrate with Whisk Studio to synchronize your recipe content for publishing, Whisk Recipes for WordPress has so much more to offer. To explore its features, read its documentation.

Installing the Plugin

The Whisk Recipes for WordPress plugin is available for free download in the WordPress plugin marketplace.

To install the plugin:

1. Download and install the **Whisk Recipes** plugin from the WordPress [marketplace](#).
2. On the Whisk's **Welcome** page, click **Let's Go** to start the setup.



3. On the **Setup Recipe URLs** page, choose a URL structure for your recipes and click **Continue**.



- If you plan to create recipes from scratch, select **Semantic URLs**. Also select a **Recipe base** to use as a slug in the URL. Here is an example of a semantic URL with preferred recipe base as **Meal Type** - <https://plugin.whisk.com/dinner/your-awesome-recipe>.
- If you already have recipes inside the WordPress posts, use **Native WordPress URLs**. Here is an example of a native WordPress URL - <https://plugin.whisk.com/postname/>.

Welcome! [Setup recipe URLs](#) [Connect to Whisk Studio](#) [Ready!](#)

Setup recipe URLs

Here you can choose URL structure for your recipes. If you are creating recipes collection from scratch, select Semantic URLs.
Otherwise, if you already have recipes inside WordPress posts, use Native WordPress URLs.

☒ Semantic URLs

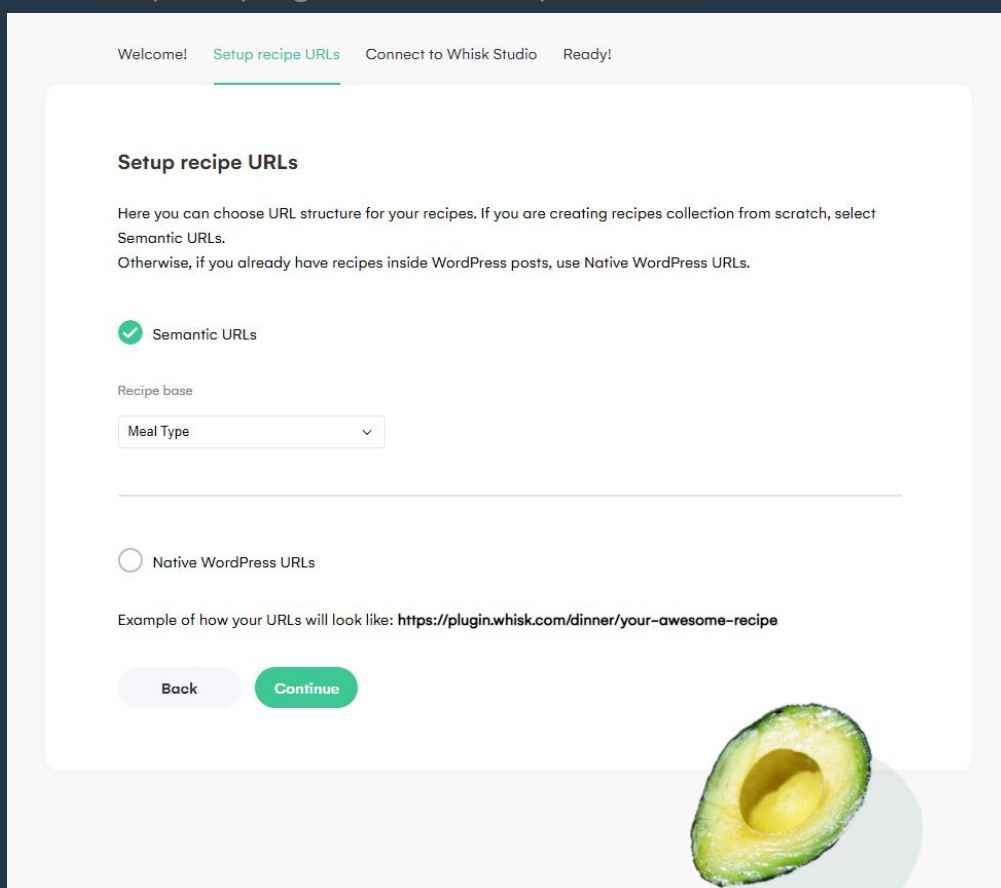
Recipe base

Meal Type ▼

☐ Native WordPress URLs

Example of how your URLs will look like: <https://plugin.whisk.com/dinner/your-awesome-recipe>

[Back](#) [Continue](#)



4. Next, connect to Whisk Studio using the **Client ID** and **Secret String**. These details can be found on the API page in Whisk Studio. For more information, see [Obtain](#)



an [API key](#) and [API Authentication](#).

Whisk

Welcome! Setup recipe URLs Connect to Whisk Studio Ready!

Connect to Whisk Studio

In order to have Nutrition Data, Auto-tagging and full Ingredient Data you must have working connection with Whisk Studio API. Create business account at <https://studio.whisk.com> and get API keys from this page <https://studio.whisk.com/api>. After that, copy and paste them into fields below and click Continue. After successful integration all new recipes should start getting enhanced data.

Nutrition Data

Calories	935.33kcal	47%
Fat	57.0g	82%
Carbs	13.14g	
Sugars	57.0g	
Protein	12.0g	
Sodium	1042.0mg	
Fiber	2.0g	

Glycemic Index: 22 Low
Glycemic Load: 1 Low

High Medium Low
7.3/10

Auto-tagging recipes

Cooking Techniques: Frying

Cuisine: Italian

Course: Burgers

Client ID

Secret

Back Continue Skip this step

Once the setup and synchronization completes, you can visit the Whisk Recipe plugin in the WordPress Dashboard to verify if all the recipes are synchronized successfully.

Note: Once the integration is enabled, any changes done in the recipe content using Whisk Studio are synchronized automatically in WordPress.



Using the Whisk Studio Sync API

When you want to use Whisk Studio for recipe content management and at the same time have your own publishing system to deploy content on your frontend, you can still have the integration done using our [Recipes Sync API](#).

How to integrate with your publishing system?

Here are the steps to set up the integration:

1. Build a connector to sync data between Whisk Studio and your publishing system.



Note: If you have the resources, you can build the custom connector on your own; else, you can outsource it to a third-party if needed. We recommend you to acquire services from [TrueLogic](#) as it is one of our trusted vendors.

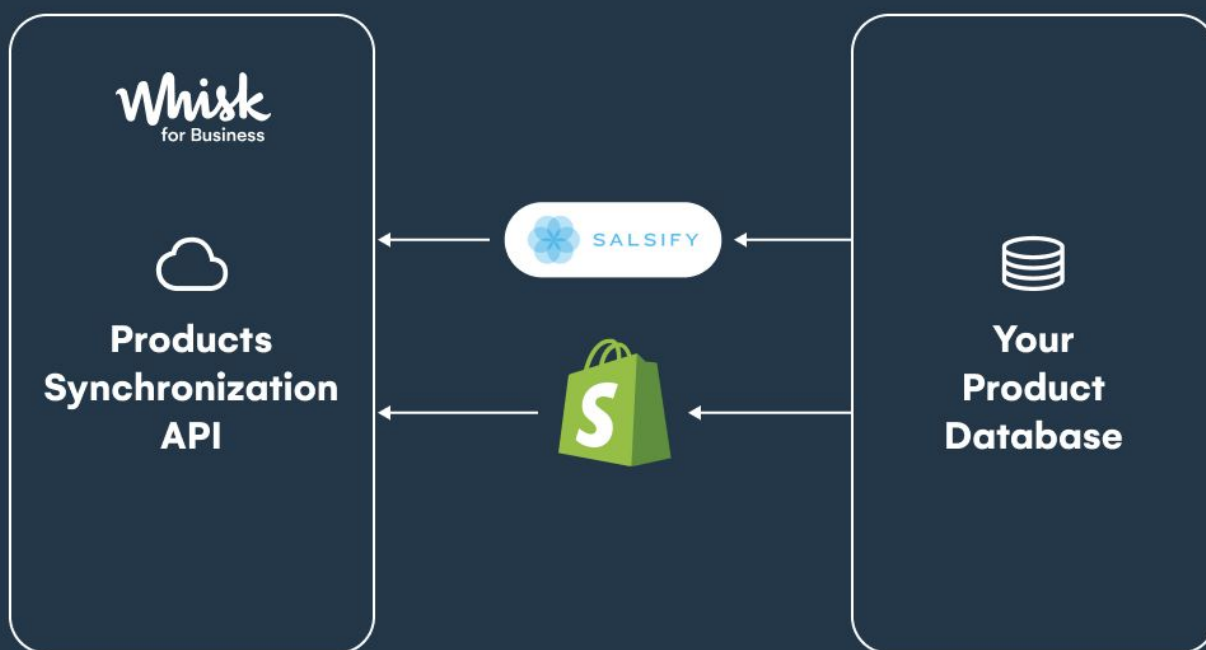
2. Use the connector to retrieve recipes from Whisk Studio using the [Recipes Sync API endpoints](#).
3. Save the metadata in your infrastructure.
4. Use your publishing system to publish the recipes on your frontend.

For more information, see [Content Synchronization using Whisk Studio Sync API](#).



Products Synchronization

Here is the general flow of how you can use Whisk Studio to manage your branded products and use them as ingredients in your recipes:



Products Synchronization Flow


There are multiple ways to synchronize your branded products in Whisk Studio:

- You can enable [integration with Salsify](#).
- You can use our [Whisk Studio app on Shopify](#).
- You can build a connector to create your Products and keep them synchronized [using our Products Sync API](#).




Using Salsify Integration

Whisk Studio supports integration with Salsify to synchronize your products' data. If you have all your products stored in Salsify, you can easily set up the integration in just a few steps and pull them into Whisk Studio. This integration is one of many ways to transfer your products into our platform.

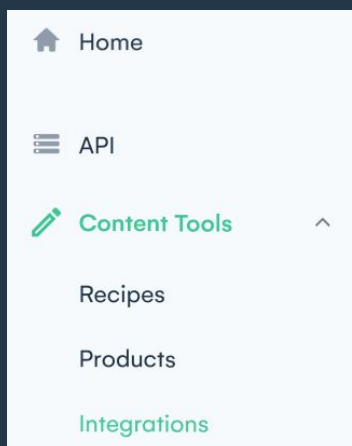
 **Note:** While integrating with Salsify, you may require us to map data for you as you may have your own set of attributes and values in your product definitions.

Follow these steps to enable the integration:

1. Log in to Whisk Studio.

 **Note:** If you are new to our platform, read the information and follow the instructions present in the [Getting Started](#) section.

2. From the left navigation panel, go to the **Integrations** page.

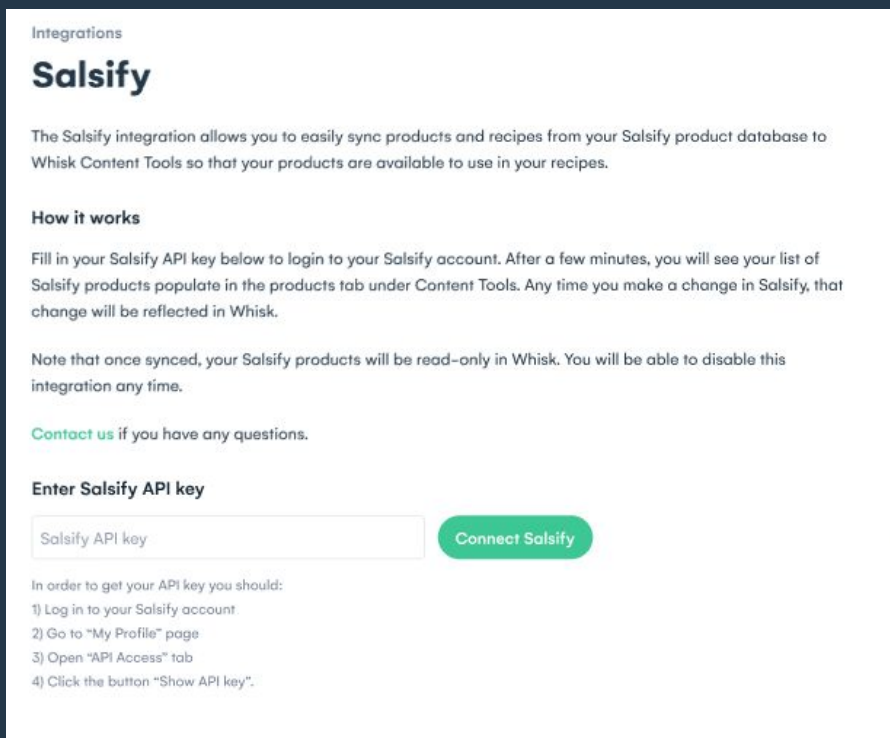


The **Integrations** page lists all the available integrations that Whisk Studio supports.

3. Click **Salsify**.



4. On the Salsify page that appears, authenticate using the **Salsify API key**. The instructions to obtain the key are present on the interface.



Integrations

Salsify

The Salsify integration allows you to easily sync products and recipes from your Salsify product database to Whisk Content Tools so that your products are available to use in your recipes.

How it works

Fill in your Salsify API key below to login to your Salsify account. After a few minutes, you will see your list of Salsify products populate in the products tab under Content Tools. Any time you make a change in Salsify, that change will be reflected in Whisk.

Note that once synced, your Salsify products will be read-only in Whisk. You will be able to disable this integration any time.

[Contact us](#) if you have any questions.

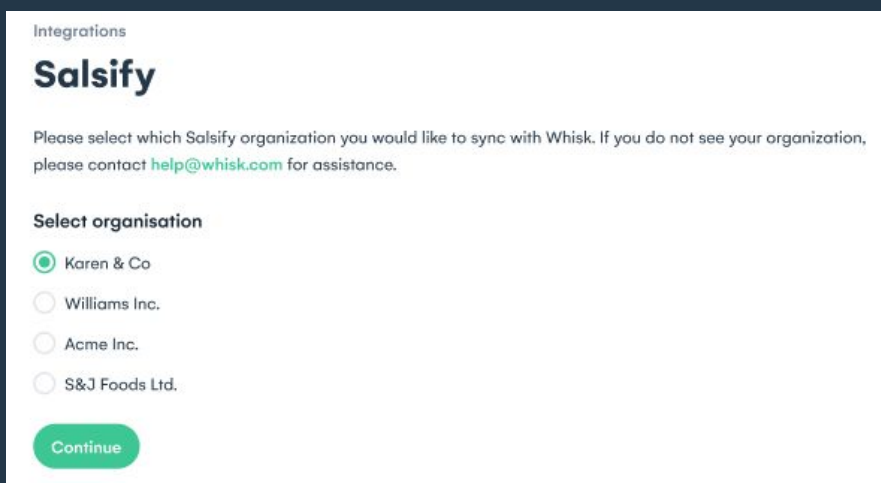
Enter Salsify API key

Connect Salsify

In order to get your API key you should:

- 1) Log in to your Salsify account.
- 2) Go to "My Profile" page
- 3) Open "API Access" tab
- 4) Click the button "Show API key".

5. Next, select the organization you want to synchronize with Whisk Studio.



Integrations

Salsify

Please select which Salsify organization you would like to sync with Whisk. If you do not see your organization, please contact help@whisk.com for assistance.

Select organisation

☒ Karen & Co

☐ Williams Inc.

☐ Acme Inc.

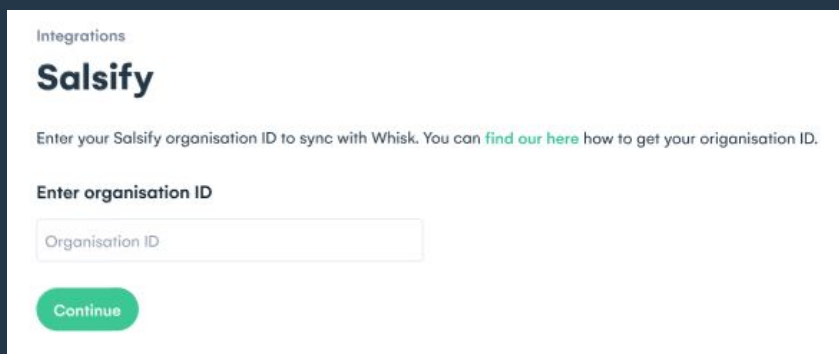
☐ S&J Foods Ltd.

Continue

If we are unable to fetch the organization id of your selected organization, you are asked to enter the Salsify **organization id** on the next page. The instructions to

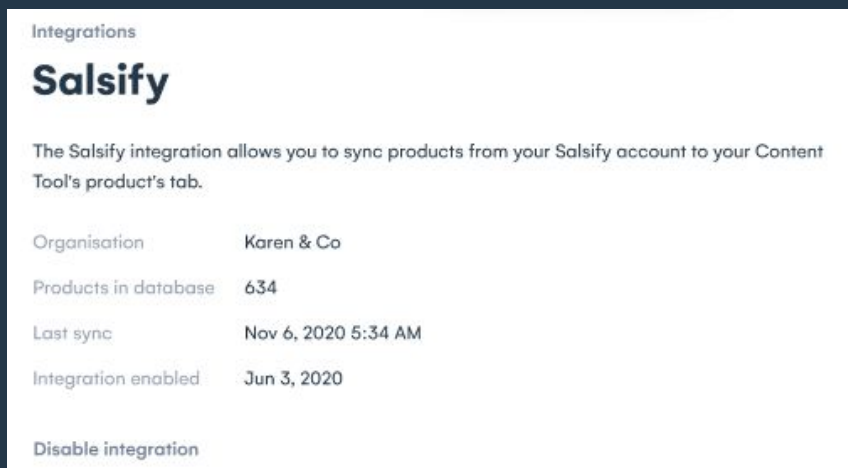


obtain the ID are present on the interface.



The screenshot shows the 'Integrations' section for 'Salsify'. It includes a heading 'Salsify', a sub-heading 'Enter your Salsify organisation ID to sync with Whisk. You can [find our here](#) how to get your organisation ID.', a label 'Enter organisation ID', a text input field with the placeholder 'Organisation ID', and a green 'Continue' button.

6. Click **Continue** to start the synchronization. As the synchronization completes, you can view the summary.



The screenshot shows the 'Integrations' section for 'Salsify' after synchronization. It includes a heading 'Salsify', a sub-heading 'The Salsify integration allows you to sync products from your Salsify account to your Content Tool's product's tab.', and a table with the following data:

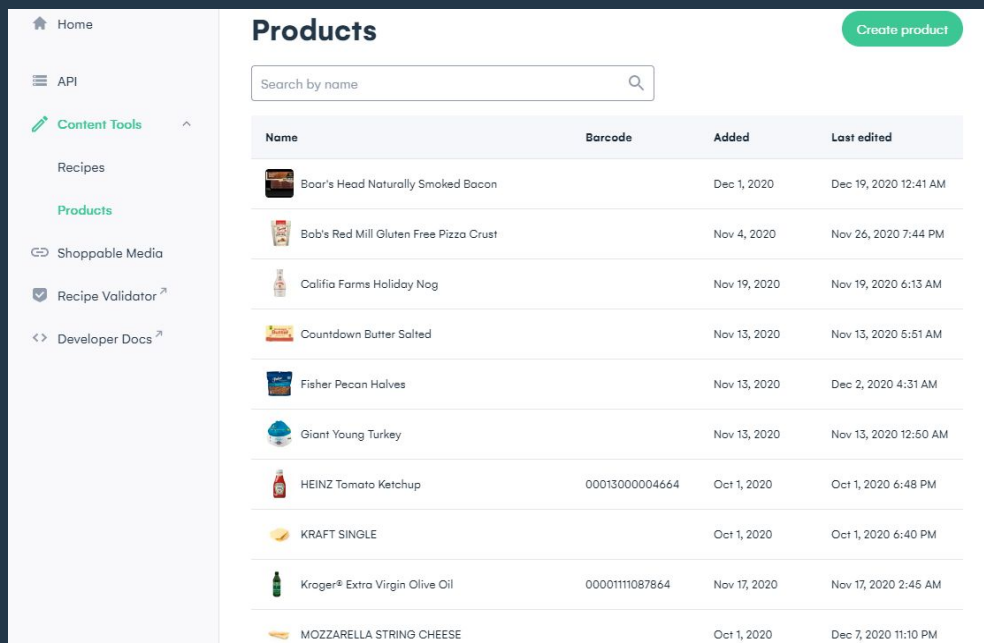
Organisation	Karen & Co
Products in database	634
Last sync	Nov 6, 2020 5:34 AM
Integration enabled	Jun 3, 2020











At the bottom, there is a link 'Disable integration'.

From the same page, you can also disable the integration at any time you want. If you face any problems with synchronization, please contact our [support](#).



You can now go to the **Products** page to see your products listed there and start using them as ingredients for your recipes.



Name	Barcode	Added	Last edited
 Boar's Head Naturally Smoked Bacon		Dec 1, 2020	Dec 19, 2020 12:41 AM
 Bob's Red Mill Gluten Free Pizza Crust		Nov 4, 2020	Nov 26, 2020 7:44 PM
 Califia Farms Holiday Nog		Nov 19, 2020	Nov 19, 2020 6:13 AM
 Countdown Butter Salted		Nov 13, 2020	Nov 13, 2020 6:51 AM
 Fisher Pecan Halves		Nov 13, 2020	Dec 2, 2020 4:31 AM
 Giant Young Turkey		Nov 13, 2020	Nov 13, 2020 12:50 AM
 HEINZ Tomato Ketchup	00013000004664	Oct 1, 2020	Oct 1, 2020 6:48 PM
 KRAFT SINGLE		Oct 1, 2020	Oct 1, 2020 6:40 PM
 Kroger® Extra Virgin Olive Oil	00001111087864	Nov 17, 2020	Nov 17, 2020 2:45 AM
 MOZZARELLA STRING CHEESE		Oct 1, 2020	Dec 7, 2020 11:10 PM



Note: Once the integration is enabled, any changes done in the products in Salsify are synchronized automatically in the Whisk Studio.


If you are new to Salsify and looking to use it as a medium to transfer your products from your infrastructure to Whisk Studio, you must first import all your products into Salsify. The following Salsify documentation links can help you understand the import process:

- Salsify Import Overview - <https://developers.salsify.com/docs/imports-overview>
- Initial Import into Salsify - <https://getstarted.salsify.com/help/the-initial-import-into-salsify>
- Salsify Import API - <https://developers.salsify.com/reference#import-format>

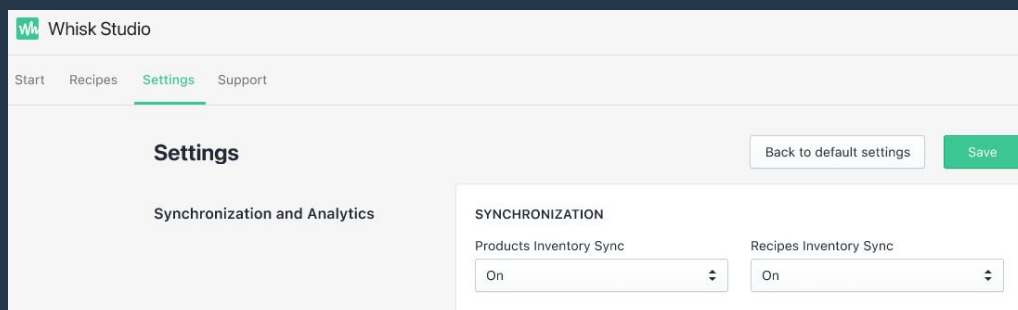


Using the Whisk Studio App on Shopify

If you choose to synchronize products while [installing the Whisk Studio app on Shopify](#), your product catalog gets synced to Whisk Studio automatically. You can start managing products on Whisk Studio without any further configuration and using them as ingredients for your recipes. For more information, see Whisk Studio's documentation.

 **Note:** Once the synchronization is enabled, you cannot edit any products pulled from Shopify to Whisk Studio. Any changes done to the products on Shopify appear automatically in Whisk Studio.

Irrespective of whether you synced the products during installation or not, you can still turn the synchronization on or off from the app's Settings page on Shopify.



Using the Whisk Studio Sync API

When you have your products stored in a database within your infrastructure or externally on some other platform, you can use our [Products Sync API](#) to set up product synchronization and create and manage products in Whisk Studio.

How to integrate with your product database?

Here are the steps to set up the integration:

1. Build a connector to sync data between Whisk Studio and your product database.

 **Note:** If you have the resources, you can build the custom connector on



your own; else, you can outsource it to a third-party if needed. We recommend you to acquire services from [TrueLogic](#) as it is one of our trusted vendors.

2. Use the connector to retrieve products from your product database, normalize the data and add them to Whisk Studio using the [Products Sync API](#).

Once the synchronization completes, you can start using your new products as ingredients in recipes on Whisk Studio. For more information, see [Content Synchronization using Whisk Studio Sync API](#).




Content Synchronization using the Whisk Studio Sync API

When you don't want to rely on any middleware app or platform for recipes and products synchronization and have the technical expertise or a connector to work with API and manage data flow, you can use our Studio Sync API to synchronize your content and manage the custom integration.

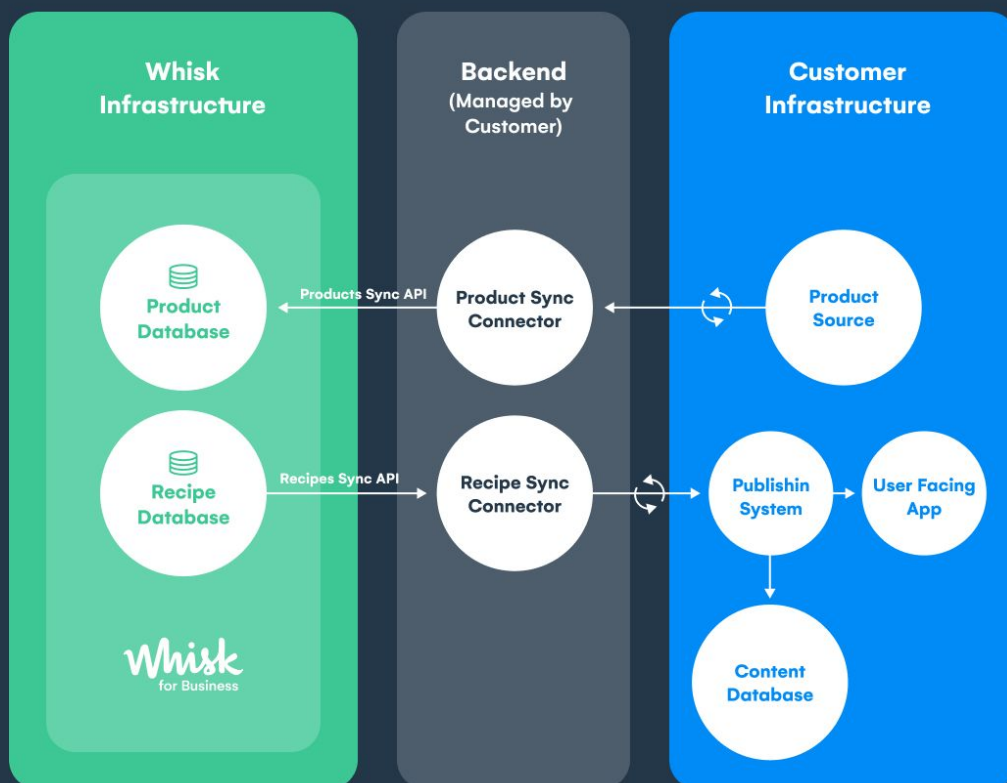
How to set up a Custom Integration with Whisk Studio?

To enable a smooth custom integration with Whisk Studio, you must follow the steps as described here.

We recommend you build a connector to sync data between Whisk Studio and your infrastructure.

 **Note:** If you have the resources, you can build the custom connector on your own; else, you can outsource it to a third-party if needed. We recommend you to acquire services from [TrueLogic](#) as it is one of our trusted vendors.





Whisk Studio Custom Integration Flow

To describe the integration process, we have split the information into two parts:

- [Products Synchronization](#)
- [Recipes Synchronization](#)

Products Synchronization

To start uploading products to Whisk Studio:

1. Obtain an API key to authenticate API usage. For more information, see [Request an API for Whisk Studio Integration](#).
2. Configure your connector or ask your developers to use the [Put Integration API endpoint](#) to enable the Custom Integration by generating an **Integration ID** with



type as **INTEGRATION_TYPE_CUSTOM**. If you already have the Custom Integration enabled, use the [Get Integrations](#) endpoint to retrieve the Integration ID.

3. Next, use the [Upsert Product Batch](#) endpoint to start uploading products on Whisk Studio. The endpoint returns a list of Product IDs that you can use to update the products later.

Once you have your Products available on Whisk Studio, you can start using them as ingredients in your recipes and then move to set up Recipes Synchronization.

Recipes Synchronization

To start fetching recipes from the platform:

1. Obtain an API key to authenticate API usage. For more information, see [Request an API for Whisk Studio Integration](#).
2. Configure your connector or ask your developers to use the [Put Integration API endpoint](#) to enable the Custom Integration by generating an **Integration ID** with type as **INTEGRATION_TYPE_CUSTOM**. If you already have the Custom Integration enabled, use the [Get Integrations](#) endpoint to retrieve the Integration ID.
3. Next, use the [Get Recipe Batch](#) endpoint to start retrieving recipes' data from Whisk Studio and save it in your database.

You must download recipes in batches filtered by the paging parameters to save time and resources. You must also save the Unix timestamp of the response and use it for filtering recipes through the **updated_at.min** paging parameter in your next request.

Now that your integration is enabled, you can configure your connector to request recipes ideally once every 5 minutes periodically.

Recipes Sync API

The Recipes Sync API provides you the ability to pull recipes individually or in a batch for data synchronization. Let's first understand the Recipe object and its data structure before looking at its endpoints' description.



Recipe Object

A Recipe object is a collection of arrays and attributes that defines the data structure for any given recipe.

This is how the data structure of a recipe looks:

```
"recipe": {
  "id": "1072689e47689e34a859633309c5d17fec8",
  "name": "Grab-and-Go Breakfast Sandwich",
  "description": "Skip the drive-thru. Your homemade breakfast sandwich, with cholesterol-free egg product, is better for you—and tastier!",
  "instructions": {
    ...
  },
  "images": [
    ...
  ],
  "source": {
    ...
  },
  "servings": 1,
  "durations": {
    ...
  },
  "ingredients": [
    ...
  ],
  "normalized_ingredients": [
    ...
  ],
  "nutrition": {
    ...
  },
  "labels": {
    ...
  },
}
```



```
"author": {  
  ...  
},  
"tips": [  
  ...  
],  
"custom_labels": [  
  ...  
],  
"language": "en",  
"updated_at_time": "1613754521094",  
"created_at_time": "1613754521094",  
"published_status": "RECIPE_PUBLISHED_STATUS_PUBLISHED"  
}
```

Core Data

The following attributes define the core of a recipe:

Attribute	Type	Description
id	string	The recipe identifier.
name	string	The full name of the recipe.
description	string	A summary describing the recipe.
servings	number	The number of people that can be served.
language	string	As per ISO 639 standards, a two-letter code indicating the language used in recipe definition.
updated_at_time	string	The last recipe update date and time in Unix time format.
created_at_time	string	The recipe creation date and time in Unix time format.



published_status	string	The recipe status. A recipe can have one of these supported status set here: <ul style="list-style-type: none"> • RECIPE_PUBLISHED_STATUS_DRAFT • RECIPE_PUBLISHED_STATUS_PUBLISHED
------------------	--------	---

In addition to the base attributes, there are objects and arrays that contain more information on the recipe.

Ingredients

This array contains information on the ingredients used in the recipe.

```
"ingredients":[
  {
    "text":"1/4 cup cholesterol-free egg product"
  },
  ...
  {
    "text":"HEINZ Tomato Ketchup",
    "linked_product":{
      "id":"5f75d73bcfed282362a8b3aa",
      "name":"HEINZ Tomato Ketchup",
      "images":[
        ...
      ]
    }
  }
],
```

Attribute	Type	Description
text	string	A specific ingredient used in the recipe — sugar 2-tbsp, flour 10g, or garlic 3-cloves.
group	string	The group name if the ingredient falls under a particular group of ingredients.



linked_product	object	This object contains details of a branded product that is stored in Whisk Studio and is being used as an ingredient.
----------------	--------	--

The `linked_product` object contains the following attributes:

Attribute	Type	Description
id	string	The product identifier.
name	string	The product's full name.
images	array	One or more images of the product. This array contains the hosted URL and size details of each product image.
external_product_id	string	An identifier assigned by the product's author.

Images

This array contains information on one or more images of the completed dish included in the recipe.

```
"images": [
  {
    "url": "https://whisk-res.cloudinary.com/image/upload/v1603979937/recipe/a322bb09c8934e66d3eca3f98c59004a.jpg",
    "responsive": {
      "url": "https://whisk-res.cloudinary.com/image/upload/v1603979937/recipe/a322bb09c8934e66d3eca3f98c59004a.jpg",
      "width": 240,
      "height": 208
    }
  },
  ...
],
```



Attribute	Type	Description
url	string	Hosted URL of an image.
responsive	object	A responsive image adjusts its size based on the screen size. This object contains attributes to access the image with flexible size based on need.

The `responsive` object contains the following attributes:

Attribute	Type	Description
url	string	Hosted URL of an image.
width	number	Image width.
height	number	Image height.

Source

This object contains details of the recipe origins.

```
"source":{
  "name":"myfoodandfamily.com",
  "display_name":"myfoodandfamily",

  "url":"https://www.myfoodandfamily.com/recipe/063765/grab-and-go-breakfast-sandwich?utm=skforu_rdp",
  "image":{

    "url":"https://whisk-res.cloudinary.com/image/upload/v1565965713/publishers/logos/myfoodandfamily-logo.png",
    "responsive":{

      "url":"https://whisk-res.cloudinary.com/image/upload/v1565965713/publishers/logos/myfoodandfamily-logo.png",
```



```

        "width":256,
        "height":256
    }
},

```

Attribute	Type	Description
name	string	Indicates the name of the recipe's origin.
display_name	string	Indicates the display name of the recipe's origin.
url	string	The recipe's origin URL.
image	object	This object includes information on the origins of the recipe's image.

Durations

This object contains information on the time it takes to prepare and cook the recipe.

```

"durations":{
  "cook_time":20,
  "prep_time":10,
  "total_time":30
},

```

Attribute	Type	Description
cook_time	number	The time (in minutes) it takes to cook the dish.
prep_time	number	The time (in minutes) it takes to prepare the items used in the recipe's instructions.



total_time	number	The total time (in minutes) it takes to prepare the dish.
------------	--------	---

Labels

This object contains information on the labels attached to the recipe that helps distinguish its type, cuisine, category, and the techniques required to cook.

```
"labels": {
  "meal_type": [
    {
      "name": "main-course",
      "display_name": "Main Course"
    },
    ...
    {
      "name": "dinner",
      "display_name": "Dinner"
    }
  ],
  "cuisine": [
    {
      "name": "french",
      "display_name": "French"
    }
  ],
  "category": [
    {
      "name": "dinner",
      "display_name": "Dinner"
    },
    {
      "name": "lunch",
      "display_name": "Lunch"
    }
  ],
  "technique": [
```



```

    {
      "name": "simmering",
      "display_name": "Simmering"
    },
    ...

    {
      "name": "pan-frying",
      "display_name": "Pan Frying"
    }
  ]
},

```

Each label attached to the recipe appears as a child-array.

Attribute	Type	Description
meal_type	array	Indicates the meal type that applies to the recipe.
cuisine	array	Indicates the recipe's cuisine.
category	array	Indicates the recipe's category.
technique	array	Indicates the cooking technique required to cook the recipe.

Each of these arrays contains the following attributes:

Attribute	Type	Description
name	string	The label name saved in the source.
display_name	string	The label name that appears on the front-end.

Author

This object includes information about the Whisk user who created the recipe.




```

"author":{
  "id": "10297e19d97dd5a43ecbd9a13172ce65bfb"
  "name":"Sara Buenfeld"
  "image":{

"url":"https://img.cjthemarket.com/images/file/author/973/20191226105710715
.jpg",
  "responsive":{

"url":"https://whisk-res.cloudinary.com/image/upload/v1602854082/author/8ba
de35e90ecf44eac6ec7ae5346d228.jpg",
    "width":300,
    "height":300
  }
},

```

Attribute	Type	Description
id	string	The recipe creator's identifier.
name	string	Author's name.
image	object	Author's image. See the images array description for more information.

Custom Labels

This array includes a list of user-defined labels linked to the recipe.

```

"custom_labels":[
  {
    "name":"brand-category",
    "labels":[
      {
        "name":"cj-chinese"
      },
      {
        "name":"cj-etc"
      }
    ]
  }
]

```



```

    }
  ],
  {
    "name": "brand",
    "labels": [
      {
        "name": "kraft-cheese"
      }
    ]
  },
  {
    "name": "barcode-number",
    "labels": [
      {
        "name": "00013000006409"
      }
    ]
  }
]

```

Attribute	Type	Description
name	string	Label group's name.
labels	array	A list of labels inside the group.

Tips

This array includes a list of advice linked to the recipe.

```

"tips": [
  {
    "header": "string",
    "text": "string"
  }
]

```



Attribute	Type	Description
header	string	A heading for the advice.
text	string	Text describing the advice.

Optional Data

The recipe data structure may include extra information based on any additional parameters included in the API request.

Normalized Ingredients

```
"normalized_ingredients":[
  {
    "text":"1 slice cooked OSCAR MAYER Bacon, cut crosswise in
half",
    "analysis":[
      {
        "product":{
          "canonical_name":"OSCAR MAYER COOKED BACON (READY
MADE)",
          "original_name":"cooked oscar mayer bacon"
        },
        "category":{
          "canonical_name":"MEATS AND SEAFOOD"
        },
        "brand":{
          "canonical_name":"OSCAR MAYER"
        },
        "quantity":1,
        "unit":"slice",
```



```

        "comment": "cut crosswise in half",
        "image_url": "https://whisk-res.cloudinary.com/image/upload/v1570185262/custom_upload/4e3434d390926ff30a3f96652529194d.jpg"
      },
      "source_text": "1 slice cooked OSCAR MAYER Bacon, cut crosswise in half"
    },
    {
      "text": "HEINZ Tomato Ketchup",
      "linked_product": {
        "id": "5f75d73bcfed282362a8b3aa",
        "name": "HEINZ Tomato Ketchup",
        "images": [
          {
            "url": "https://whisk-res.cloudinary.com/image/upload/v1600195675/inventory_item/20c81dcc9b003de383b1891828b59fc5.jpg"
          }
        ]
      },
      "source_text": "HEINZ Tomato Ketchup"
    },
    ...
  ],

```

Attribute	Type	Description
text	string	A specific ingredient used in the recipe — sugar 2-tbsp, flour 10g, or garlic 3-cloves.
group	string	The group name if the ingredient falls under a particular group of ingredients.



linked_product	object	This object contains details of a branded product that is stored in Whisk Studio and is being used as an ingredient. For more information, see the ingredients array .
analysis	array	Detailed information on the ingredient that is not linked to any product stored in Whisk Studio.
source_text	string	The ingredient name saved in the source.

The **analysis** array includes the following attributes:

Attribute	Type	Description
product	object	The name of the product being used as an ingredient. It should not be confused with the linked_product object as specific recipes may specify a branded product as an ingredient, which may not be one of their own branded products on Whisk Studio.
category	object	The name of the category to which the ingredient belongs.
brand	object	The ingredient product's brand name.
quantity	number	The ingredient volume or count.
comment	string	Any additional info attached with the ingredient.
unit	string	The measurement unit of the ingredient's quantity.
image_url	string	The source location of the ingredient's image.

Instructions

This object includes information on each step required in the recipe preparation.

```
"instructions":{
  "steps":[
    {
      "text":"Cook egg product in skillet sprayed with cooking spray on
medium heat 3 min. or until set, stirring occasionally.",
```



```

      "images": [
        {
          "url": "https://whisk-res.cloudinary.com/image/upload/v1603979936/recipe/33c0f4a5c00651929f57d00b398aafda.jpg",
          "width": 424,
          "height": 640
        }
      ],
    },
    {
      "text": "Fill muffin halves with egg product, Singles and bacon.",
      "images": [
        {
          "url": "https://whisk-res.cloudinary.com/image/upload/v1603979947/recipe/62848f0ee14fd5808e541e3e29934e72.jpg",
          "width": 424,
          "height": 640
        }
      ]
    }
  ]
},

```

Attribute	Type	Description
steps	array	An array containing a list of steps required to cook the recipe.

The `steps` array contains the following attributes:

Attribute	Type	Description
text	string	A specific step in the list of instructions to cook the recipe — “Heat the oil in a large frying pan”.



group	string	The group name if the step falls under a particular group of steps.
images	array	One or more images attached to the step. This array contains the hosted URL and size details of each image.
custom_labels	array	One or more user-defined labels attached to the step. For more information, see the custom_labels array.

Nutrition

This object contains information about the recipe's nutritional value.

```
"nutrition":{
  "status":"STATUS_AVAILABLE",
  "total":[
    {
      "label":"Magnesium",
      "code":"NUTRITION_CODE_MG",
      "value":82.61800000000001,
      "unit":"NUTRITION_UNIT_MG"
    },
    ...
  ],
  "coverage": 0.93,
  "health_score":{
    "value":6.687124447756711,
    "nutrients_influence":[
      {
        "code":"NUTRITION_CODE_FAT_UNSAT",
        "influence":0.7762592808559045,
        "comment":"Strong positive impact"
      },
      ...
    ]
  },
  "glycemic_index":{
    "value":42.12
  },
}
```



```
"glycemic_load":{
  "value":7.66
},
```

Attribute	Type	Description
status	string	The recipe's nutritional information availability status.
total	array	This array contains a detailed analysis of all nutrients available in the recipe.
coverage	string	A value between 0-1 to indicate the extent of analysis done while calculating the nutritional facts.
health_score	array	The recipe's health score.
glycemic_index	object	The glycemic index is a value that indicates the recipe's impact on blood glucose levels.
glycemic_load	object	The glycemic load is a value that indicates the recipe's impact on blood sugar levels.

Each object contains additional attributes.

Total

This array contains information about the nutrients available in the recipe. Each nutrient carries the following information:

Attribute	Type	Description	Example
code	enum	The nutrient code. You can find the list of all supported nutrient codes here .	NUTRITION_CODE_MG
label	string	The name of the nutrient.	Magnesium
value	integer	The nutrient's value/amount.	82.61800000000001



unit	enum	The measurement unit of the nutrient's value.	<ul style="list-style-type: none"> • NUTRITION_UNIT_G • NUTRITION_UNIT_MG • NUTRITION_UNIT_MKG • NUTRITION_UNIT_KCAL
------	------	---	--

Health Score

This array contains information on the health score assigned to the recipe.

Attribute	Type	Description
value	double	The health score assigned to the recipe.
nutrients_influence	array	The components based on which the health score is derived.

Glycemic Index

This object includes information on the glycemic index score assigned to the recipe. It helps in understanding the impact of the recipe on blood sugar levels.

Attribute	Type	Description
value	double	The glycemic index score.

Glycemic Load

This object includes information on the glycemic load score assigned to the recipe. It helps in understanding the impact of the recipe on blood glucose levels.

Attribute	Type	Description
value	double	The glycemic load score.



Deleted Recipe Object

Any recipes that you delete from Whisk Studio are marked as deleted and they appear as a deleted recipe when you pull recipes in batch using the [Get Recipe Batch](#) endpoint.

A deleted recipe object has the following data structure:

```
"deleted_recipe": {
  "id": "107e25b932ca5a2484e8727e9e7cad8b2ca",
  "deleted_at_time": "1610627213000",
  "created_at_time": "1603990751339"
}
```

The `deleted_recipe` object contains the following attributes:

Attribute	Type	Description
id	string	The recipe identifier.
deleted_at_time	string	The recipe deletion date and time in Unix time format.
created_at_time	string	The recipe creation date and time in Unix time format.

Endpoints

Recipe API includes two endpoints that you can periodically use to pull recipe data from Whisk Studio.

Get Recipe


This endpoint retrieves recipe information using the recipe identifier.

Base URL	https://api.studio.whisk.com/recipes/v1/get
----------	---



Query Parameters

You can append the following query parameters to the base URL to pull data of any particular recipe from Whisk Studio:

Parameter	Type	Description	Example
recipe_id*	string	The recipe identifier.	1072689e47689e34a859633309c5d17fec8
fields	array	Any additional recipe details to retrieve. <div>  Note: For performance reasons, we recommend using this parameter only when required. </div>	<ul style="list-style-type: none"> EXTRA_RECIPE_FIELD_INVALID EXTRA_RECIPE_FIELD_NORMALIZED_INGREDIENTS EXTRA_RECIPE_FIELD_INSTRUCTIONS EXTRA_RECIPE_FIELD_NUTRITION EXTRA_RECIPE_FIELD_INGREDIENTS_LINKED_PRODUCTS EXTRA_RECIPE_FIELD_AUTO_LABELS
integration_id*	string	An identifier to indicate the integration type. See Integration API for more information.	3a2b8cfd-27cd-4c81-ab79-e31e03e56c20

Sample Curl Request

```
curl -X GET
"https://api.studio.whisk.com/recipes/v1/get?recipe_id=1072689e47689e34a859633309c5d17fec8&fields=EXTRA_RECIPE_FIELD_NORMALIZED_INGREDIENTS&fields=EXTRA_RECIPE_FIELD_INSTRUCTIONS&fields=EXTRA_RECIPE_FIELD_NUTRITION&fields=EXTRA_RECIPE_FIELD_INGREDIENTS_LINKED_PRODUCTS&fields=EXTRA_RECIPE_FIELD_AUTO_
```



```
LABELS&integration_id=3a2b8cfd-27cd-4c81-ab79-e31e03e56c20"
-H "Accept: application/json" \
-H "Authorization: Token <YOUR-API-ACCESS-TOKEN>"
```

Sample Response

This is how a successful response looks like. To understand its data structure, see [Recipe Object](#).

200: OK

```
{
  "recipe": {
    "recipe": {
      "id": "1072689e47689e34a859633309c5d17fec8",
      "name": "Grab-and-Go Breakfast Sandwich",
      "description": "Skip the drive-thru. Your homemade breakfast sandwich, with cholesterol-free egg product, is better for you--and tastier!",
      "instructions": {
        "steps": [
          {
            "text": "Cook egg product in skillet sprayed with cooking spray on medium heat 3 min. or until set, stirring occasionally."
          },
          {
            "text": "Fill muffin halves with egg product, Singles and bacon."
          }
        ]
      },
      "images": [
        ...
      ],
      "source": {
        ...
      },
      "servings": 1,
    }
  }
}
```



```

    "durations": {
      "prep_time": 10,
      "total_time": 10
    },
    "ingredients": [
      {
        "text": "1/4 cup cholesterol-free egg product"
      },
      ...
    ],
    "normalized_ingredients": [
      ...
    ],
    "nutrition": {
      ...
    },
    "labels": {
      ...
    },
    "author": {
      "id": "10255a34128905f4869a27b5bdf24519820"
    },
    "language": "en",
    "updated_at_time": "1610730959413",
    "created_at_time": "1610725156807",
    "published_status": "RECIPE_PUBLISHED_STATUS_PUBLISHED"
  "custom_labels": [
    ...
  ]
}
}
}

```

Get Recipe Batch


This endpoint retrieves a list of recipes filtered by pagination and timestamp.



Base URL	<code>https://api.studio.whisk.com/recipes/v1/get/batch</code>
-----------------	--

Query Parameters

You can append the following query parameters to the base URL to pull data from Whisk Studio:

Parameter	Type	Description	Example
<code>paging.limit</code>	integer	The maximum number of recipes to retrieve.	3
<code>paging.after</code>	string	This parameter is used for cursor based pagination. It takes a recipe ID as its value and retrieves only those recipes that appear after it in the database.	1070a06032934f84e788a73cd58c017366b4
<code>fields</code>	array	Any additional recipe details you want to pull for each recipe in the batch. <div>  Note: For performance reasons, we recommend using this parameter only when required. </div>	<ul style="list-style-type: none"> EXTRA_RECIPE_FIELD_INVALID EXTRA_RECIPE_FIELD_NORMALIZED_INGREDIENTS EXTRA_RECIPE_FIELD_INSTRUCTIONS EXTRA_RECIPE_FIELD_NUTRITION EXTRA_RECIPE_FIELD_INGREDIENTS_LINKED_PRODUCTS EXTRA_RECIPE_FIELD_AUTO_LABELS
<code>updated_at.min</code>	string	Date and time in Unix time format to indicate that only those recipes get retrieved	1514788200000



		that were updated after it.	
updated_at.max	string	Date and time in Unix time format to indicate that only those recipes get retrieved that were updated before it.	1610999038495
integration_id*	string	An identifier to indicate the integration type. See Integration API for more information.	3a2b8cfd-27cd-4c81-ab79-e31e03e56c20

Sample Curl Request

```
curl -X GET
"https://api.studio.whisk.com/recipes/v1/get/batch?paging.limit=3&paging.after=1070a06032934f84e788a73cd58c017366b4&fields=EXTRA_RECIPE_FIELD_NORMALIZED_INGREDIENTS&fields=EXTRA_RECIPE_FIELD_INSTRUCTIONS&fields=EXTRA_RECIPE_FIELD_NUTRITION&fields=EXTRA_RECIPE_FIELD_INGREDIENTS_LINKED_PRODUCTS&fields=EXTRA_RECIPE_FIELD_AUTO_LABELS&updated_at.max=1610999038495&integration_id=3a2b8cfd-27cd-4c81-ab79-e31e03e56c20"
-H "Accept: application/json" \
-H "Authorization: Token <ACCESS-TOKEN>"
```

Sample Response

This is how a successful response looks like. To understand its data structure, see [Recipe Object](#).

200: OK

```
{
  "recipes": [
    {
      "recipe": {
        "id": "107218d97a8c9294971802be6c8b81bf137",
```



```

    "name": "Laurel's Malaysian Inspired Pavlova",
    ...
    "language": "en",
    "updated_at_time": "1607041141957",
    "created_at_time": "1607041141957",
    "published_status": "RECIPE_PUBLISHED_STATUS_DRAFT"
  }
},
{
  "recipe": {
    "id": "1072689e47689e34a859633309c5d17fec8",
    "name": "Grab-and-Go Breakfast Sandwich",
    "description": "Skip the drive-thru. Your homemade breakfast sandwich, with cholesterol-free egg product, is better for you--and tastier!",
    ...
    "language": "en",
    "updated_at_time": "1610730959413",
    "created_at_time": "1610725156807",
    "published_status": "RECIPE_PUBLISHED_STATUS_DRAFT"
    ...
  }
},
{
  "recipe": {
    "id": "1072bd455bd5b5b438cb3654036673fb45e",
    "name": "Pudding-Chocolate Morsel Cookies",
    "description": "Check out these delicious Pudding-Chocolate Morsel Cookies! Made with chocolate pudding mix and loaded with chocolate chips, these morsel cookies stay chewy even in the cookie jar.",
    ...
    "language": "en",
    "updated_at_time": "1607136379241",
    "created_at_time": "1607112817836",
    "published_status": "RECIPE_PUBLISHED_STATUS_PUBLISHED"
  }
}

```




```
}  
],  
"paging": {  
  "total": "3",  
  "after": "2bd455bd-5b5b-438c-b365-4036673fb45e"  
}  
}
```

Products Sync API

Product Synchronization API offers you the ability to synchronize your branded products in Whisk Studio and use them as ingredients in your recipes. Let's first understand the Product object and its data structure before looking at its endpoints' description.

Product Object

A Product object is a collection of arrays and attributes that defines the data structure for any given product.

This is how the data structure of a product looks:

```
{  
  "product": {  
    "id": "5febbcd1bc31826090025de5",  
    "name": "Bitchin' Sauce Original",  
    "gtin": "08801007763910",  
    "description": "The OG. Where Bitchin' began. Creamy lemon and  
garlic. Simple and Satisfying.",  
    "brand": "Bitchin'",  
    "created_at": "1609284817642",  
    "updated_at": "1609284863384",  
  }  
}
```



```

    "images": [
      ...
    ],
    "language": "en",
    "price": {
      ...
    },
    "instructions": [
      ...
    ],
    "ingredients": [
      ...
    ],
    "url": "https://bitchinsauce.com/",
    "author": {
      ...
    },
    "amount": {
      ...
    },
    "serving_size": {
      ...
    },
    "nutrition": [
      ...
    ],
    "external_product_id": "30252563"
  }
}

```

Core Data

The following attributes define the core of the product:

Attribute	Type	Description
-----------	------	-------------



id	string	The product identifier.
name	string	The product's full name.
gtin	string	The <i>Global Trade Item Number</i> assigned to the product.
description	string	A summary describing the product.
brand	string	The product's brand name.
created_at	string	The product creation date and time in Unix time format
updated_at	string	The last product update date and time in Unix time format
language	string	As per ISO 639 standards, a two-letter code indicating the language used in product definition.
url	string	The product's origin URL.
external_product_id	string	An identifier assigned by the product's author.

In addition to the base attributes, there are some objects and arrays that contain more information on the product.

Images

This array contains information on one or more images of the product.

```
"images":[
  {
    "url":"https://img.cjthemarket.com/images/file/product/973/20191226105710715.jpg",
    "responsive":{
```



```

"url": "https://whisk-res.cloudinary.com/image/upload/v1602854082/inventory_
item/8bade35e90ecf44eac6ec7ae5346d228.jpg",
  "width": 1200,
  "height": 1200
},
...
],

```

Attribute	Type	Description
url	string	The product's image URL.
responsive	object	A responsive image adjusts its size based on the screen size. This object contains attributes to access the image with flexible size based on need.

Responsive

This object contains the following attributes:

Attribute	Type	Description
url	string	Hosted URL of an image.
width	number	Image width.
height	number	Image height.

Price

This object contains information on the product cost.

```

"price": {
  "rrp": {
    "currency_code": "USD",
    "cents": "500",

```



```
}
},
```

Attribute	Type	Description
rrp	object	It contains attributes related to the recommended retail price of the product.

Recommended Retail Price (rrp)

Attribute	Type	Description
currency_code	string	The three-letter currency code as per ISO 4217 standards.
units	string	The product's price in cents. For example, if the currency_code is USD, then 100 cents makes 1 USD.

Instructions

This array contains information on the product instructions.

```
"instructions": [
  {
    "text": "Store in the refrigerator.",
    "group": "Storage"
  }
  ...
],
```

Attribute	Type	Description
text	string	Instruction's text.
group	string	Instruction's group.



Ingredients

This array contains information on the product ingredients.

```
"ingredients":[
  {
    "text":"Water, Grapeseed Oil, Almonds, Lemon Juice, Soy Sauce,
    Nutritional Yeast, Garlic, Spices, Sea Salt",
    "group":"Consolidated"
  },
  ...
],
```

Attribute	Type	Description
text	string	Ingredient's description.
group	string	Ingredient's group.

Author

This object contains information on the Whisk user who created the product.

```
"author":{
  "id":"ehd678gkttgfr58zd",
  "name":"Dan",
  "image":{

    "url":"https://img.cjthemarket.com/images/file/author/973/20191226105710715
    .jpg",
    "responsive":{

      "url":"https://whisk-res.cloudinary.com/image/upload/v1602854082/author/8ba
      de35e90ecf44eac6ec7ae5346d228.jpg",
```



```

        "width":300,
        "height":300
      }
    }
  },

```

Attribute	Type	Description
id	string	The product creator's identifier.
name	string	The product creator's name.
image	object	The product creator's image. See the images array description for more information.

Amount

This object contains information on the product's quantity.

```

"amount":{
  "unit":"g",
  "quantity":200
}

```

Attribute	Type	Description
unit	string	The measurement unit of the product's quantity.
quantity	number	The product volume or count.

Serving Size

This object contains information on the number of portions that can be served by the product.



```
"serving_size":{
  "unit":"g",
  "quantity":200
},
```

Attribute	Type	Description
unit	string	The measurement unit of the product's serving.
quantity	number	The serving volume or count.

Optional Data

The product data structure may include extra information based on any additional parameters included in the API request.

Nutrition

This array contains information about the product's nutritional value.

```
"nutrition":[
  {
    "label":"Energy",
    "code":"NUTRITION_CODE_ENERC_KCAL",
    "value":"1200"
    "unit":"NUTRITION_UNIT_KCAL"
  },
  ...
],
```

Attribute	Type	Description	Example
label	string	The name of the nutrient.	Energy
code	enum	The nutrient code. You can find the list of all supported nutrient codes here .	NUTRITION_CODE_ENERC_KCAL



value	number	The nutrient's value/amount.	1200
unit	enum	The measurement unit of the nutrient's value.	<ul style="list-style-type: none">• NUTRITION_UNIT_G• NUTRITION_UNIT_MG• NUTRITION_UNIT_MKG• NUTRITION_UNIT_KCAL

Endpoints

Product API includes three endpoints that you can periodically use to pull, create, update, or delete products.

Get Product

This endpoint retrieves product information using the product identifier.


Base URL	https://api.studio.whisk.com/products/v1/get
----------	---

Query Parameters

You can append the following query parameters to the base URL to pull data of any particular product from Whisk Studio:

Parameter	Type	Description	Example
product_id*	string	The product identifier.	5fb83b7cdc54e606d155613e



fields	array	Any additional details to retrieve. <div> Note: For performance reasons, we recommend using this parameter only when required.</div>	Product_Extra_Field_Nutrition
integration_id*	string	An identifier to indicate the integration type. See Integration API for more information.	a8dd7887-f752-4783-8724-16467a6665ef

Sample Curl Request

```
curl -X GET
"https://api.studio.whisk.com/products/v1/get?product_id=5fb83b7cdc54e606d155613e&fields=EXTRA_PRODUCT_FIELD_NUTRITION&integration_id=a8dd7887-f752-4783-8724-16467a6665ef"
-H "Accept: application/json" \
-H "Authorization: Token <YOUR-API-ACCESS-TOKEN>"
```

Sample Response

This is how a successful response looks like. To understand its data structure, see [Product Object](#).

200: OK



```

{
  "product": {
    "id": "5fb83b7cdc54e606d155613e",
    "name": "Sabra Classic Guacamole",
    "description": "It's always \"fiesta o'clock\" with the fresh taste of
our guacamole, made with Mexican-grown Hass avocados. Our authentic recipe
is flavorful and chunky--just like homemade.",
    "created_at": "1605909372473",
    "updated_at": "1605909372473",
    "images": [
      {
        "url": "https://img.cjthemarket.com/images/file/product/973/2019122610571071
5.jpg",
        "responsive": {
          "url":
"https://whisk-res.cloudinary.com/image/upload/v1605909200/v3/user-recipes/
it1mlzs2ab34tdhfsfxk.png",
          "width": 640,
          "height": 640
        }
      }
    ],
    "language": "en",
    "ingredients": [
      {
        "text": "Hass Avocado, Onion, Tomato, Jalapeno Pepper, Salt,
Garlic, Lime Juice Concentrate, Dehydrated Onion, Cilantro, Ascorbic Acid
(added to maintain freshness).\"
      }
    ],
    "url": "https://sabra.com/dips/guacamole/classic-guacamole.html",
    "nutrition": [
      {
        "label": "Energy",
        "code": "NUTRITION_CODE_ENERC_KCAL",
        "unit": "NUTRITION_UNIT_KCAL"
      },
      . . .
    ]
  }
}

```



```
}
}
```


Get Product Batch

This endpoint retrieves a list of products filtered by pagination and timestamp.

Base URL	<code>https://api.studio.whisk.com/products/v1/get/batch</code>
-----------------	---

Query Parameters

You can append the following query parameters to the base URL to pull data from Whisk Studio:

Parameter	Type	Description	Example
<code>paging.limit</code>	integer	The maximum number of products to retrieve.	3
<code>paging.after</code>	string	This parameter is used for cursor based pagination. It takes a product ID as its value and retrieves only those products that appear after it in the database.	5ff905b3030cc211abb31fb7
<code>fields</code>	array	Any additional product details you want to pull for each product in the batch. <div>  Note: For performance reasons, we recommend using this parameter only when required. </div>	Extra_Product_Field_Nutrition



updated_at.min	string	Date and time in Unix time format to indicate that only those products get retrieved that were updated after it.	1608886602824
updated_at.max	string	Date and time in Unix time format to indicate that only those products get retrieved that were updated before it.	1610999038495
integration_id*	string	An identifier to indicate the integration type. See Integration API for more information.	a8dd7887-f752-4783-8724-16467a6665ef

Sample Curl Request

```
curl -X GET
"https://api.studio.whisk.com/products/v1/get/batch?paging.limit=3&fields=EXTRA_PRODUCT_FIELD_NUTRITION&updated_at.max=1610999038495&integration_id=3a2b8cfd-27cd-4c81-ab79-e31e03e56c20"
-H "Accept: application/json" \
-H "Authorization: Token <ACCESS-TOKEN>"
```

Sample Response

This is how a successful response looks like. To understand its data structure, see [Product Object](#).

200: OK

```
{
  "products": [
    {
      "id": "5febbcd1bc31826090025de5",
```



```

    "name": "Bitchin' Sauce Original",
    "gtin": "00123456789121",
    "description": "The OG. Where Bitchin' began. Creamy lemon and
garlic. Simple and Satisfying.",
    "created_at": "1609284817642",
    "updated_at": "1609284863384",
    ...
  },
  {
    "id": "5ff905b3030cc211abb31fb7",
    "name": "Heinz Ketchup",
    "created_at": "1610155443277",
    "updated_at": "1610155443277",
    ...
  },
  {
    "id": "5fc68566968f2e51ce557699",
    "name": "Boar's Head Naturally Smoked Bacon",
    "created_at": "1606845798758",
    "updated_at": "1608318685438",
    ...
  }
],
"paging": {
  "total": "15",
  "after": "3"
}
}

```

Upsert Product Batch

This endpoint lets you use the **upsert** method to create or update products in a batch. It can be instrumental when you want to make changes to the products or create new products in bulk, especially when you want to have a [custom integration](#) in place with Whisk Studio.

Base URL	<code>https://api.studio.whisk.com/products/v1/upsert/batch</code>
-----------------	--



Body Parameter

You can specify the following body parameter to create or update products in Whisk Studio:

Parameter	Type	Description	Example
body*	object	A body object contains multiple product definitions. In each definition, the <code>external_product_id</code> determines whether to update or create a new product. If the identifier already exists in the database, the product information gets updated, and if not, then a new product gets generated.	See the Body Object Example below.

Body Object Example

```
{
  "batch":[
    {
      "request_id":"1",
      "payload":{
        "external_product_id":"551d2f0b-fba9-43ac-9a4b-7737e53823cf",
        "name":"UpdateProduct",
        "description":"Updated product description",
        "brand":"Whisk",
        "language":"en"
      }
    },
    {
      "request_id":"2",
```



```

    "payload":{
      "external_product_id":"551d2f0b-fba9-43ac-9a4b-7737e53823c1",
      "name":"NewProduct",
      "description":"New product",
      "brand":"Whisk",
      "language":"en"
    }
  ],
  "integration_id":"3a2b8cfd-27cd-4c81-ab79-e31e03e56c20"
}

```

The body object contains the following attributes that you can specify to create or update products:

Attribute	Type	Description	Example
batch	array	It contains a group of definitions to update or create products.	<pre> { "batch": [{ "request_id": "1", "payload": { ... } }, { "request_id": "2", "payload": { ... } }], } </pre>
integration_id	string	An identifier to indicate the integration type. See Integration API for more information.	3a2b8cfd-27cd-4c81-ab79-e31e03e56c20



The **batch** object contains the following attributes:

Attribute	Type	Description	Example
request_id	string	An identifier for each product data submission request.	"request_id": "2",
payload	object	It contains individual product definitions for either data creation or update. To understand the data structure of a product definition, see the Product Object .	"payload":{ "external_product_id":"551d2f0b-fba9-43ac-9a4b-7737e53823cf", "name":"UpdateProduct", "description":"Updated product description", "brand":"Whisk", "language":"en" }

Sample Curl Request

```
curl -X POST "https://api.studio.whisk.com/products/v1/upsert/batch"
-H "Accept: application/json" \
-H "Authorization: Token <ACCESS-TOKEN>"
-H "Content-Type: application/json"
-d "{ \"batch\":[ { \"request_id\":\"1\", \"payload\":{
  \"external_product_id\":\"551d2f0b-fba9-43ac-9a4b-7737e53823cf\",
  \"name\":\"UpdateProduct\", \"description\":\"Updated product
description\", \"brand\":\"Whisk\", \"language\":\"en\" } }, {
  \"request_id\":\"2\", \"payload\":{
  \"external_product_id\":\"551d2f0b-fba9-43ac-9a4b-7737e53823c1\",
  \"name\":\"NewProduct\", \"description\":\"New product\",
  \"brand\":\"Whisk\", \"language\":\"en\" } } ],
  \"integration_id\":\"3a2b8cfd-27cd-4c81-ab79-e31e03e56c20\"}"
```



Sample Response

This is how a successful response looks.

200: OK

```
{
  "results": [
    {
      "success": {
        "request_id": "1",
        "product_id": "5ff46349642e29bec3731026",
        "external_product_id": "551d2f0b-fba9-43ac-9a4b-7737e53823cf"
      }
    },
    {
      "success": {
        "request_id": "2",
        "product_id": "5ff4671f642e29bec3777d76",
        "external_product_id": "551d2f0b-fba9-43ac-9a4b-7737e53823c1"
      }
    }
  ]
}
```

Delete Product Batch

This endpoint lets you delete products in a batch. It can be really useful when you want to remove multiple products in bulk.

Base URL	https://api.studio.whisk.com/products/v1/delete/batch

Body Parameter

This endpoint supports the body parameter that you can specify to delete products from Whisk Studio:



Parameter	Type	Description	Example
body*	object	This body object contains a list of product identifiers to delete the related products.	<pre>{ "product_ids": ["5fe5a94f642e29bec369c986", "5fe5a94e642e29bec369c92f"], "integration_id": "e07032d4-ab74-4736-ae84-57091fae6210" }</pre>

Sample Curl Request

```
curl -X DELETE "https://api.studio.whisk.com/products/v1/delete/batch"
-H "Accept: application/json" \
-H "Authorization: Token <ACCESS-TOKEN>"
-H "Content-Type: application/json"
-d "{ \"product_ids\": [ \"5fe5a94f642e29bec369c986\",
  \"5fe5a94e642e29bec369c92f\" ], \"integration_id\":
  \"e07032d4-ab74-4736-ae84-57091fae6210\"}"
```

Sample Response

This is how a successful response looks like. To understand its data structure, see [Product Object](#).

200: OK

```
{}
```



Integrations API

The Integrations API provides you the ability to manage your Whisk Studio Integrations. Let's first understand the Integration object and its data structure before looking at its endpoints' description.

Integrations Object

An Integrations object is a collection of attributes that store details of each integration.

This is how the data structure of an Integrations definition looks:

```
{
  "integrations": [
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_CUSTOM",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1608910441740",
      "updated_at": "1608910441740"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_SALSIFY",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1609768210609",
      "updated_at": "1609846831662"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_SHOPIFY",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1611242960379",
      "updated_at": "1611242960379"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_WORDPRESS",
      "status": "INTEGRATION_STATUS_ENABLED",
```



```

    "created_at": "1611346856098",
    "updated_at": "1611346856098"
  }
]
}

```

The following attributes define the core of an integration object:

Attribute	Type	Description
id	string	The integration identifier. The Whisk Studio Sync API needs this identifier to authorize the API calls.
type	enum	Indicates the type of integration. It supports the following values: <ul style="list-style-type: none"> • <code>INTEGRATION_TYPE_CUSTOM</code> • <code>INTEGRATION_TYPE_SALSIFY</code> • <code>INTEGRATION_TYPE_SHOPIFY</code> • <code>INTEGRATION_TYPE_WORDPRESS</code>
status	enum	Indicates the current status of Integration. It supports the following values: <ul style="list-style-type: none"> • <code>INTEGRATION_STATUS_DISABLED</code> • <code>INTEGRATION_STATUS_ENABLED</code>
updated_at	string	The last integration update date and time in Unix time format.
created_at	string	The integration creation date and time in Unix time format.

Endpoints

Integrations API includes three endpoints that you can use to manage different integration types in your Whisk Studio account.



Put Integration

This endpoint offers you the ability to enable a new integration type in Whisk Studio.

Base URL	<code>https://api.studio.whisk.com/integrations/v1/add</code>
-----------------	---

You can specify the following body parameter to enable a new integration type:

Parameter	Type	Description	Example
<code>body*</code>	object	<p>This body object contains the integration type that you want to enable in Whisk Studio. The supported integration types are:</p> <ul style="list-style-type: none"> <code>INTEGRATION_TYPE_CUSTOM</code> <code>INTEGRATION_TYPE_SALSIFY</code> <code>INTEGRATION_TYPE_SHOPIFY</code> <code>INTEGRATION_TYPE_WORDPRESS</code> 	<pre>{ "type": "INTEGRATION_TYPE_CUSTOM" }</pre>

Sample Curl Request

```
curl -X PUT "https://api.studio.whisk.com/integrations/v1/add"
-H "Accept: application/json" \
-H "Authorization: Token <YOUR-API-ACCESS-TOKEN>"
-d "{ \"type\": \"INTEGRATION_TYPE_CUSTOM\"}"
```

Sample Response

This is how a successful response looks. To understand its data structure, see [Integration Object](#).



200: OK

```
{
  "integration": {
    "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "type": "INTEGRATION_TYPE_CUSTOM",
    "status": "INTEGRATION_STATUS_ENABLED",
    "created_at": "1608910441740",
    "updated_at": "1608910441740"
  }
}
```

Get Integrations

This endpoint offers you the ability to retrieve details of all integration types that you had enabled in your Whisk Studio account.

Base URL	<code>https://api.studio.whisk.com/integrations/v1/get</code>
-----------------	---

You can use just the base URL and the authentication header to pull details of all Integration types linked to your Whisk Studio account.

Sample Curl Request

```
curl -X GET "https://api.studio.whisk.com/integrations/v1/get"
-H "Accept: application/json" \
-H "Authorization: Token <YOUR-API-ACCESS-TOKEN>"
```

Sample Response

This is how a successful response looks. To understand its data structure, see [Integration Object](#).



200: OK

```
{
  "integrations": [
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_CUSTOM",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1608910441740",
      "updated_at": "1608910441740"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_SALSIFY",
      "status": "INTEGRATION_STATUS_DISABLED",
      "created_at": "1609768210609",
      "updated_at": "1609846831662"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_SHOPIFY",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1611242960379",
      "updated_at": "1611242960379"
    },
    {
      "id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
      "type": "INTEGRATION_TYPE_WORDPRESS",
      "status": "INTEGRATION_STATUS_ENABLED",
      "created_at": "1611346856098",
      "updated_at": "1611346856098"
    }
  ]
}
```

Update Integration

This endpoint lets you update the integration status on a specific integration type.



Base URL	<code>https://api.studio.whisk.com/integrations/v1/update/state</code>
-----------------	--

Body Parameters

You can specify the following body parameter to update the status:

Parameter	Type	Description	Example
<code>body*</code>	object	This body object contains the integration identifier and the attribute to update the status.	<pre>{ "integration_id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx", "status": "INTEGRATION_STATUS_DISABLED" },</pre>

Sample Curl Request

```
curl -X POST "https://api.studio.whisk.com/integrations/v1/update/state"
-H "Accept: application/json" \
-H "Authorization: Token <ACCESS-TOKEN>"
-H "Content-Type: application/json"
-d "{ \"integration_id\": \"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx\",
  \"status\": \"INTEGRATION_STATUS_DISABLED\"},"
```

Sample Response

This is how a successful response looks.

200: OK

```
{}
```



Whisk's Partners API Integration

As an out-of-the-box approach, in addition to using our Whisk Studio Sync API to retrieve and synchronize your recipes, you can also use our Whisk's Partners API to make use of many other functionalities that the API provides. This approach is definitely for you if you are looking to build an app or a website related to shoppable recipe content.

Here are the following advantages of using Partners API:

- There is no need to store recipe content in your infrastructure.
- You can provide your users with visual delight and smarter personalized experiences.
- You can let your users closely connect with the Whisk ecosystem.
- Your app or a website can have a flexible set of functionalities and look and feel.

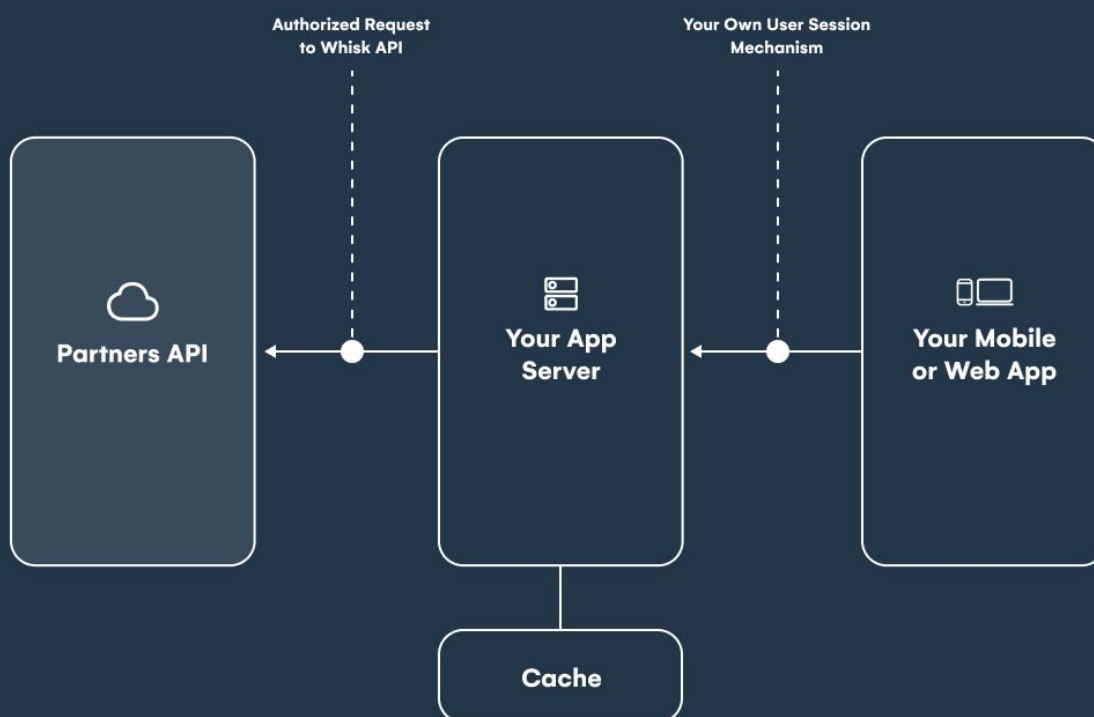
Whisk's Partners API lets you use some of our pre-designed functionalities based on your content like recipe search, recipe feed, meal planning, shopping list, etc. You can find the full list of APIs [here](#).



Supported Integration Methods

You can integrate with Whisk's Partners API on either server-side or on client-side based on your requirements.

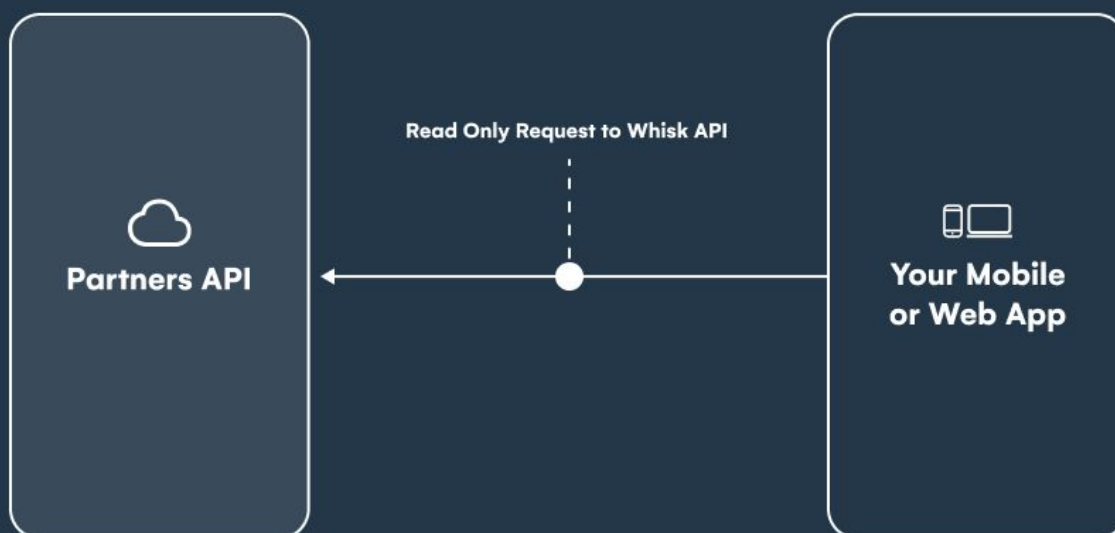
Server-Side Integration



If you want your users to have a personalized and responsive experience, we generally recommend you to have a server layer between Whisk's Partners API and your client application. This layer allows you to have your user session mechanism with your client application. We also recommend having a cache on your server-side for non-personalized requests to avoid exceeding API limits. To have this integration done, you can use [OAuth](#) and [Server Token](#) for API authentication.



Client-Side Integration




The client-side integration lets you directly integrate your client application with Whisk but have limited access. For this integration type, you can use the [Client Token](#) for API authentication to call read-only API endpoints like Get Recipe or Recipe Search. While you choose this integration type, you should know that you can exceed the API limits with excessive usage, and it may result in your client application getting stopped.

Integration Process

To get started with Whisk's Partners API integration is simple.

1. Get a Sandbox API key. For more information, see [Obtain an API key](#).
2. Choose an integration approach that suits your requirements. See [Integration Methods](#) for more information.
3. Try and test our [API](#).

 **Note:** Whisk has its OpenAPI Specs published on Swagger to let you try out the API calls directly in your browser. You can call our API [here](#).


4. When you are ready to go with this approach, request production API access to start building your app or a website. For more information, see [Obtain an API key](#).



Partners API Integration Sample

Whisk Playground is a live demo application that you can refer to see how integration with Whisk looks.

URL	https://showcase.whisk.com/
-----	---

 **Note:** This demo showcases just a basic layout and a few general functionalities. We provide more functionalities through our API, which you can use to improve the user experience of your application or website.



Working with Whisk API

API Authentication

You can access the Whisk Studio Synchronization API and Whisk's Platform API using the following authentication mechanisms:

User Access Token

A user access token is required for a personalized experience when a Whisk user context is necessary to access a resource. It is generated using the OAuth 2.0 flow where you need to obtain a Client ID/Secret and register redirect URLs.

Server Token

A server token is used to access data with a non-personalized experience. It only provides you read-only access to the API. It should be stored securely and used only from servers. The API key that is provided to you in Whisk Studio is a server token that allows you to enable supported integrations and authenticate Synchronization API access. For more information, see [Obtain an API key](#).

Client Token

A client ID is a public identifier for apps. You can embed it into your native mobile binaries or desktop apps to identify your app while using our API. It offers you a non-personalized experience.



Sandbox API Token

When you register with us, you receive a key for our Whisk API sandbox called the Test Kitchen that emulates our production API's behavior and helps you get started with the Whisk Partners API features. Since we provide it to you for testing, it retrieves only dummy data and has some limitations.



Note: You can find more information on each authentication mechanism [here](#).

API Querying

- Some of our Sync API endpoints provide cursor-based pagination. A cursor is a random string that points to a specific item in a list of data. A string pointed to an element can be changed in the future. Therefore, your app should not store cursors.
- When using the Recipe API to pull recipes, mostly in custom integration scenarios, we recommend using the **fields** parameter in your query only when required for performance reasons.

API Documentation

To make your experience integrating with us more convenient and better, it would help if you familiarize yourself with our API documentation. Visit our documentation [here](#).

Error Handling

The API endpoints may return the following error codes:

Error Code	Response	Description
400: Bad Request	<pre>{ "error_code": "REAL_CODES_ARE_IN_ENDP OINT_DESCRIPTION",</pre>	This failed response appears when error codes are found in the endpoint query.



	<pre>"message": "Additional details about error are not static and can be changed" }</pre>	
401: Unauthorized	<pre>{ "code": "auth.tokenNotFound" }</pre>	<p>This failed response appears due to API authentication failure. The possible error codes that may appear are:</p> <ul style="list-style-type: none"> • <code>auth.tokenNotFound</code> • <code>auth.tokenExpired</code> • <code>auth.tokenInvalid</code> • <code>auth.tokenRequired</code>
404: Not Found	<pre>{ "error_code": 0, "message": "<Entity> with id <entity_id> has been deleted" }</pre>	<p>This failed response appears when you try to read a deleted entity.</p>
500: Internal Server Error	<p>This is unexpected response, something is wrong on our side, please contact: help@whisk.com</p>	<p>This failed response appears when something is not right on Whisk's end. Please send a message to help@whisk.com, and be sure to include both the Request and Response data. We'll get back to you soon.</p>



We wish you to have the best experience integrating with Whisk!

