Import API

The Import API provides a programmatic way to insert or update large amounts of data in Account Engagement. It uses Account Engagement's existing API structures, patterns, and terminology. The Import API automates the existing prospect import feature set. Only prospect import is supported currently.

IMPORTANT The user's role must have the Prospect Import ability to start an API import. For more information, see Account Engagement User Roles

Use the Import API to insert or update large sets of data, when you don't require synchronous completion responses, or when batch upsert limitations are too restrictive. Currently, only Prospect upsert is supported.

The Import API processes many records asynchronously in batches. Each batch requires a minimum amount of system resources to run, so larger batches are more efficient. Small batches may result in slower performance.

The Import API lets you import a CSV file of prospects. Columns in the CSV correspond to Account Engagement field names. Rows correspond to prospects to be upserted. Each column must match a valid field name, or validation fails and the CSV is rejected.

When a row in the CSV file matches an existing prospect, the prospect is updated. Field values in the CSV, including blank (null) values, overwrite existing values for the prospect. To modify this default behavior, see the `columns` param on the Create endpoint.

A new prospect is created when a matching prospect isn't found. All standard and custom fields are supported.

In API Version 3, prospects are matched by email address. If the matched prospect is in the recycle bin, that record won't upsert unless the `restoreDeleted` option is specified during import creation. The rest of the import isn't affected when a record is skipped.

An import contains a set of records divided into one or more batches of data. A batch is a set of records sent to the server in an HTTP POST request. The import specifies which object is processed and what type of operation is used. The Import API currently supports only the Prospect object and the Upsert operation.

Batches are processed in parallel, and batches are subdivided into smaller groups of objects for processing.

The order in which individual records, batches, and entire imports are processed isn’t guaranteed.

The Import resource is used to create an import, get status for an import, upload data as a batch, and change status for an import.

When a batch finishes processing, the result for failed records is available in a result set resource. All other records are assumed to be successful.

Users create, submit, and retrieve results of imports with the following steps.

Create an import that specifies object and action.

Upload data to the server in one or more batches.

To submit the import for processing, set the state of the import to "Ready". After the import is submitted, you can't add more batches of data or abort the import.

Check the status of the import at a reasonable interval. We recommend that you wait a few minutes between calls. Calls to check import status count against API call limits. When the results of the status check indicate a complete import, the results also contain statistics for creates, updates, and failures.

If there are failures, download a log of failures. The log includes only records not inserted or updated.

Imports left in the "Open" state expire after 24 hours.

Each account can process 1000 batches per day.

Each batch must be smaller than 10 MB.

The daily data limit is 10 GB.

Each import can contain up to 10 batches.

The daily Account Engagement API call limit and the concurrent Account Engagement API call limit apply to Import API calls just as they would any other Account Engagement API calls.

Imports expire:

24 hours after creation if the import isn't submitted. No records are imported in this case, even if batches of data have been added.

7 days after completion.

After an import expires, it can't be changed and attempts to check its status or retrieve error results will fail.

This document assumes that you're familiar with connecting to the Account Engagement API, managing prospects, and creating CSV files. You can import data by using these endpoints.

Used to create a new asynchronous import. If a batch of import data is included in this request, the import can be submitted for processing as part of this operation. Otherwise one or more batches must be created and associated with this import using the do/batch endpoint and the import must be submitted for processing with the do/update endpoint.
ContentType: application/json or multipart/form-data
Use multipart/form-data if sending prospect data in the create request. Use application/json if subsequent do/batch requests are made with prospect data.
Input Representation (sent as "**importInput**" if ContentType is multipart/form-data):
operation: The operation to be executed. Currently only “Upsert” is available. See Import Operation enum for more information.
object: The current object to be run the import against. Currently only “Prospect” is available. See Import Object enum for more information.
restoreDeleted: (Optional) If a record within the CSV file matches a record that was moved to the Recycle Bin, usually through a Delete, the record can be restored and updated. If the property is true, then the record is restored and updated. If false, the record is ignored and an error occurs The error appears in the error document after the import is completed. Default if not specified is false.
state: (Optional) Specify "Ready" to indicate that the CSV file included in the request contains all prospects for this import. In this case, do/batch and do/update requests aren't required and should be omitted.
columns: (Optional) If specified, the columns must include the same number of fields as the CSV file does. If they don't match then a **400 BAD_REQUEST** failure occurs. The elements for this are:
field: (Required) The field name must match one of the columns in the CSV file
overwrite: (Optional, default true) When set to true and updating an existing record, the value in the input modifies ("overwrite") the existing value in the database. When set to false, the existing value in the database won’t update regardless of the value of the input. If the input row is a "Create", then this option is ignored and the value is present in the database.
nullOverwrite: (Optional, default true) When set to true and updating an existing record, if the value in the input is an empty string or a string containing any number of whitespaces then the value in the database is set to empty string or null. When set to false the empty input value is ignored and the existing value in the database isn't updated. If the input row is a "Create" then this option is ignored and the empty string or null is present in the database.
A single part with the name "**importFile**" contains the CSV file for the batch. The file must contain a header row.
Status Code: 200
Output Representation
id: The ID of the import.
state: The state of the import, which should always be returned as “Open” or "Waiting". "Waiting" is only returned if the "Ready" state is specified in the input. See Import State enum.
isExpired: Indicates that the import has expired. After an import expires, it can’t be changed, and attempts to check its status or retrieve error results will fail.
batchesRef: (Optional) The full URL path to add batches of data to the import. This is included in the response only if the import remains in the "Open" state.
Status code: 4xx
Error codes: See Error Codes & Messages

Allows adding batches of data to an existing import when in the "Open" state.
{id}: The ID of the import.
ContentType: multipart/form-data
A single part with the name "**importFile**" should contain the CSV file for the batch. The file should contain a header row.
Column names must match field names. For example, to set campaign, pass “campaign_id”. Columns that don't match existing field names cause validation to fail on this step. Each batch must contain an identical header with the same fields in the same order.
Status Code: 204
Output Representation: Empty Body
Status code: 4xx
Error codes: See Error Codes & Messages

Used to submit the import by changing the state to "Ready". After this step, no more batches of data can be added, and processing of the import begins.

{id}: The ID of the import.
ContentType: application/json
Status Code: 200
Status code: 4xx
Error codes: See Error Codes & Messages

Returns the current state of the import. If processing is complete, the output provides a path to the results of the operation along with any statistics about the operation.
{id}: The ID of the import.
Status Code: 200
Output Representation
id: The ID of the import.
state: The current state of the import. See Import State enum.
isExpired: Indicates that the import expired. After an import expires, it can't be changed, and attempts to check its status or retrieve error results will fail.
When import state is “Open”:
batchesRef: The full url path to add batches of data to the import.
When processing is complete:
createdCount: Count of prospects created.
updatedCount: Count of prospects updated.
If **isExpired** is false:
If there are errors:
errorsRef: The full URL path to retrieve errors CSV.
errorCount: Count of error operations.
Status code: 4xx
Error codes: See Error Codes & Messages

Used by administrators to retrieve a list of imports and their status. A user must have the “Admin > Imports > View” ability to execute this endpoint.
created_after: (Optional) Filters the results to return only imports created after the specified time.
created_before: (Optional) Filters the results to return only imports created before the specified time.
updated_after: (Optional) Filters the results to return only imports updated after the specified time.
updated_before: (Optional) Filters the results to return only imports updated before the specified time.
status: (Optional) Filters the results to return imports in the given state. Allowed values are those of the Import State enum. If not specified, all statuses are returned.
origin: (Optional) Filters the result to return only imports initiated from the wizard or through the api, or both, The possible values are: "ui", "api", "all"
sort_by: (Optional) Sorts the results by the specified property value. Allowed values are "id", "created_at", or "updated_at". If not specified, the results are sorted by "created_at".
sort_order: (Optional) Used with `sort_by`. Adjusts the direction of the sort using the values "ascending" or "descending". If not specified, the results are in "descending" order.
limit: (Optional) Specifies the number of results to be returned. Default value: 20. ***Note:*** This number can't be larger than 100.
offset: (Optional) Specifies the first matching record according to the specified sorting order to be returned in the query response. The first offset matching record is omitted from the response. Default value: 0. Example: Specifying offset=400 returns the results starting with the 401st record matched by the provided criteria.
format: (Optional) Specifies the output format generated by the api. The possible values are "json" or "xml", when not specified it defaults to "xml".
For date representation in the parameters above the value can be `today`, `yesterday`, `last_7_days`, `this_month`, `last_month`, or a custom time specified in GNU Date Input Syntax format.
Status Code: 200
Output Representation
result: A collection of imports
total_results: The total number of results in the output.
import: A collection of imports. If there are no results, this property is omitted. If there's a single result, it’s an object. If there are multiple results, it’s an array of results.

id: The ID of the import. This ID is used to check the status of the import.
state: The state of the import. Displays "Waiting" when the import is queued for processing, "Processing" when the server is working on the import and "Complete" when the import has completed. See Import State enum.
origin: How the import was initiated. See Import Origin enum.
isExpired: Indicates that the import expired. After an import expires, `true` is returned and no data associated with the import can be downloaded.
createdAt: The date and time the import is created in the timezone of the user making the request.
updatedAt: The date and time the import is updated in the timezone of the user making the request.
The import representation returned in query doesn't contain `resultRefs`. Use the read endpoint for the import to get the full import representation.
Status Code: 4xx
Error codes: See Error Codes & Messages

Download errors associated with the specified import (after it's complete).
{id}: The ID of the import.
CSV data with error info for any rows that failed to result in inserts or updates. For error descriptions, see Prospect Import Errors.
Status code: 4xx
Error codes: See Error Codes & Messages

"Open" : The import is created and can have batches added
"Ready" : All imports are added and ready to be processed
"Waiting" : The import is waiting to be picked up by a processing agent.
"Processing" : The import batches are being processed
"Complete" : All batches completed and results are ready to be consumed
"Upsert" : Upsert objects
"Prospect" : Operate on Prospect
"UI" : The import request is initiated through the wizard
"API" : The import request is initiated through an api call