Sublogin Web API

Sublogin Documentation

Guide for Magento administrators, technical project owners, and implementation teams.

LAST UPDATED
2026-10-13

SOURCE
Open this article online

EXTENSION
Magento 2 Extension Sublogin

Sublogin Web API

Install `mageb2b/sublogin-api` when an ERP, identity workflow or administration service must create and maintain sublogin records through Magento's REST or SOAP layer. The package exposes sublogin record CRUD. Role, group and permission records have their own API package: `mageb2b/sublogin-role-api` (see Role API). Budgets and approvals do not expose API routes.

Install and authorize

```
composer require mageb2b/sublogin-api
php bin/magento module:enable MageB2B_SubloginApi
php bin/magento setup:upgrade
```

Create a dedicated Magento integration and grant only the Sublogin resources it needs:

Operation	Route	ACL resource
Read one	`GET /V1/sublogin/:id`	`MageB2B_Sublogin::manage`
Search	`GET /V1/sublogin/search`	`MageB2B_Sublogin::manage`
Create	`POST /V1/sublogin`	`MageB2B_Sublogin::save`
Update	`PUT /V1/sublogin/:id`	`MageB2B_Sublogin::save`
Delete	`DELETE /V1/sublogin/:id`	`MageB2B_Sublogin::delete`

Use HTTPS and store the integration token outside source code and logs.

Identity and scope fields

Field	Meaning
`id`	Sublogin record ID. The update and delete routes use this value.
`entity_id`	Parent Magento customer entity ID.
`website_id`	Magento website used for email uniqueness and account scope.
`store_id`	Store context used by the sublogin and, depending on configuration, email templates.
`email`	Sublogin login address. It must not collide with a customer or sublogin in the same account-sharing scope.

Field	Meaning
`firstname`, `lastname`, `prefix`	Person details.
`active`, `expire_date`	Access state and optional expiry date.
`send_backendmails`, `is_subscribed`, `create_sublogins`	Email, newsletter and delegated-management flags.
`customer_address_ids`	Parent customer address IDs assigned to the sublogin.
`addresses`	Sublogin-owned address objects.
`default_billing`, `default_shipping`	Default sublogin address IDs.

The data interface also contains internal reset-token and last-login fields. Do not send, copy or log reset tokens in integrations.

Fields added by other packages

The base package and add-ons expose additional data through Magento extension attributes rather than the primary Sublogin payload:

Attribute	On	Added by
`role_id`, `group_id`	`SubloginInterface`	`mageb2b/sublogin-role`
`order_needs_approval`, `order_approval_amount_check`	`SubloginInterface`	`mageb2b/sublogin-orderapproval`
`sales_report_config`	`SubloginInterface`	base package (stores the user's report column visibility)
`sublogin_id`	`OrderInterface`, `CartInterface`, `AddressInterface`	base package
`sublogin_approval_token`, `sublogin_approver_ids`	`OrderInterface`	`mageb2b/sublogin-orderapproval`
`can_create_sublogins`, `max_number_sublogins`, `sublogin_optional_email`	`CustomerInterface`	base package
`can_manage_role`	`CustomerInterface`	`mageb2b/sublogin-role`

Read these through the object's `extension_attributes` in REST responses. The customer-level attributes are admin-managed: customer-side writes that submit them through `custom_attributes` are rejected — see Protected Customer Attributes.

Sublogin API tokens

API authentication with sublogin credentials issues an `oauth_token` bound to that sublogin identity (`sublogin_id` plus the parent `customer_id`). The token keeps the sublogin's permissions during REST requests — it is not equivalent to the parent customer's token. Tokens are invalidated when the sublogin is deactivated, expires or is deleted, so a removed buyer cannot keep calling the API.

Address shape

An address supports name, company, street, city, country, region, postcode, telephone and default flags. In this API contract, `street` is one string, not Magento's usual array of street lines.

Create a sublogin

Replace the IDs and secrets with values from your staging system:

```
curl --request POST "https://store.example/rest/V1/sublogin" \
  --header "Authorization: Bearer <integration-token>" \
  --header "Content-Type: application/json" \
  --data '{
    "sublogin": {
      "entity_id": 123,
      "website_id": 1,
      "store_id": 1,
      "email": "elena.fischer@northstar-industrial.example",
      "password": "<new-password>",
      "firstname": "Elena",
      "lastname": "Fischer",
      "active": true,
      "customer_address_ids": [456],
      "addresses": [
        {
          "firstname": "Elena",
          "lastname": "Fischer",
          "company": "Northstar Industrial Supplies GmbH",
          "street": "Werkstraße 18",
          "city": "Düsseldorf",
          "country_id": "DE",
          "postcode": "40210",
          "telephone": "+49 211 5550100",
          "default_shipping": true,
          "default_billing": false
        }
```

```
      ]
    }
  }'
```

The password is validated against Magento's customer password policy and stored as a hash. Prefer a secure account-setup flow when an external system should not know the user's final password.

Search with Magento search criteria

```
curl --get "https://store.example/rest/V1/sublogin/search" \
  --header "Authorization: Bearer <integration-token>" \
  --data-urlencode
"searchCriteria[filterGroups][0][filters][0][field]=entity_id" \
  --data-urlencode
"searchCriteria[filterGroups][0][filters][0][value]=123" \
  --data-urlencode "searchCriteria[pageSize]=50" \
  --data-urlencode "searchCriteria[currentPage]=1"
```

Paginate every synchronization. Do not assume one response contains the full company account.

Update and delete safely

Use `PUT /V1/sublogin/:id` with a complete, validated `sublogin` payload. Read the current record first when your integration does not own every field.

Before `DELETE`, decide how historic orders should display after the person is removed. Deactivation is safer when access must end but the identity should remain available for audit and reporting.

Error checklist

- **401 or 403:** verify the integration token and the route's exact ACL resource.
- **Email already exists:** confirm `website_id` and Magento account-sharing scope, then search both customers and sublogins.
- **Password rejected:** use the Magento password policy configured for the store.
- **Address rejected:** send `street` as a string and provide the required country, postcode and telephone fields.
- **Record found in the wrong website:** always supply and filter by the intended website scope.

Related: Budget Integration Boundary and Import and Export.