Middleware requirements

Creatio SCIM Middleware Implementation Requirements

Date: 2026-05-19

This document describes how a customer-owned middleware layer must call Creatio SCIM endpoints to provision users and groups from an external identity provider (IdP).

The intended architecture is:

```
Customer IdP -> Customer middleware -> Creatio SCIM API


```

The middleware is the SCIM client from Creatio's point of view. It must send requests to Creatio that are compatible with the behavior previously validated with Microsoft Entra provisioning.

If the middleware also exposes SCIM endpoints to the customer IdP, implement the same resource and discovery surface described in this guide on the inbound side, then translate each operation to the Creatio-facing request shown here. The simplest compatible design is to return Creatio SCIM resource IDs from the middleware as the inbound SCIM `id` values, because the IdP will later use those IDs in `PATCH`, `PUT`, `GET`, and `DELETE` calls. If the middleware uses its own IDs, it must maintain a reliable mapping to Creatio SCIM IDs.

This guide is self-contained for implementation teams. It explains authentication, discovery, supported resources, every supported user and group use case, expected request bodies, expected Creatio behavior, error handling, and acceptance tests.

1. Implementation Principles

The middleware must follow these principles:

1. Use Creatio SCIM endpoints only. Do not provision users, contacts, roles, or memberships by direct database access, OData, or other Creatio APIs.
2. Treat the Creatio SCIM `id` as the resource identifier after creation. Store the returned `id` in the middleware mapping table.
3. Use the IdP's stable immutable identifier as `externalId` when available.
4. Send user membership changes using Creatio SCIM user IDs, not IdP user IDs, email addresses, or usernames.
5. Use group membership operations on `PATCH /Groups/{id}` as the primary membership flow, because that is the closest Entra-style provisioning pattern.

6. Always send `active` in user create and full update payloads. If omitted, the current DTO default can be interpreted as `false`.
7. Use only supported SCIM attributes and filters listed in this document.
8. Expect Creatio group-to-role provisioning to require administrator action after groups are received. SCIM receives groups first; Creatio roles are mapped or created from the SCIM Administration page.

2. Creatio Setup Prerequisites

Before middleware development starts, the Creatio administrator must provide these values from the SCIM profile authorization setup:

- Token endpoint.
- SCIM base URL.
- OAuth client ID.
- OAuth client secret.
- OAuth scope or scopes value, if configured.

The SCIM profile must be active. The associated technical user and its OAuth integration must also be active. If no active SCIM profile is linked to the OAuth technical user, Creatio rejects provisioning requests.

Use the exact SCIM base URL shown by Creatio. Depending on the environment it can include the workspace segment, for example:

```
https://example.creatio.com/0/rest/scim/v2


```

or:

```
https://example.creatio.com/rest/scim/v2


```

Do not build this URL manually if Creatio provides it in the SCIM profile modal.

3. Authentication

Creatio SCIM endpoints require OAuth 2.0 bearer authentication.

3.1 Token Request

Use client credentials against the Creatio token endpoint.

Example:

```
POST {TOKEN_ENDPOINT}
Content-Type: application/x-www-form-urlencoded

grant_type=client_credentials
&client_id={CLIENT_ID}
&client_secret={CLIENT_SECRET}
&scopes={SCOPES}


```

Some Creatio environments or OAuth clients may use `scope` instead of `scopes`. Use the parameter name provided by the Creatio administrator or the SCIM profile setup instructions.

Token lifetime in the current implementation is 3600 seconds. Middleware must track token expiry and refresh before the token expires.

3.2 SCIM Request Headers

Every SCIM request must include:

```
Authorization: Bearer {ACCESS_TOKEN}
Accept: application/scim+json
Content-Type: application/scim+json


```

`application/json` is generally accepted by the current implementation, but `application/scim+json` is preferred for SCIM clients.

Refresh the token before expiry. If Creatio returns `401`, acquire a new token and retry once. If the retry also returns `401`, stop and report a configuration error.

4. Common SCIM Conventions

The SCIM projection layer is Creatio's internal SCIM directory. It stores the IdP-facing state for users, groups, and group memberships. SCIM GET responses are built from this projection layer. A projection user or projection group can exist before it is materialized into a real Creatio user or role. Projection IDs are the SCIM resource IDs returned to middleware.

4.1 Resource IDs

Creatio returns a GUID string as the SCIM `id`.

Example:

```
{
  "id": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
  "externalId": "idp-user-123",
  "userName": "alex.lee@example.com"
}


```

The middleware must store this mapping:

```
IdP user id or externalId -> Creatio SCIM User id
IdP group id or externalId -> Creatio SCIM Group id
```

Group member references must use the Creatio SCIM User `id`, for example:

```
{
  "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
  "type": "User"
}
```

Do not send IdP IDs, usernames, or emails in `members.value`.

4.2 Pagination

List endpoints support:

```
startIndex
count
```

Rules:

- `startIndex` is 1-based.
- Default `startIndex` is `1`.
- Default `count` is `100`.
- Maximum normalized `count` is `200`.
- If `count=0`, Creatio returns no resources but still returns `totalResults`.

Example:

```
GET {SCIM_BASE_URL}/Users?startIndex=1&count=100
```

4.3 Attribute Projection

List endpoints support:

```
attributes
excludedAttributes
```

Rules:

- Do not send both in the same request.

- `schemas`, `id`, and `meta` are always returned.
- Entra-style requests often use `excludedAttributes=members` for group listing. The middleware can use this to reduce payload size.

Example:

```
GET {SCIM_BASE_URL}/Groups?
filter=displayName%20eq%20%22Sales%22&excludedAttributes=members

```

4.4 Filtering

Filtering is supported on `GET /Users` and `GET /Groups`.

Use filters mainly for lookup before create or update.

Supported user filter attributes:

- `userName`
- `emails.value`
- `active`
- `externalId`
- `meta.created`
- `meta.lastModified`
- `title`
- `preferredLanguage`

Supported group filter attributes:

- `displayName`
- `externalId`
- `meta.created`
- `meta.lastModified`

Supported operators in the current implementation:

- `eq`
- `pr`
- `gt`
- `ge`
- `lt`
- `le`

- `and`
- `or`

Recommended Entra-compatible lookup filters:

```
GET {SCIM_BASE_URL}/Users?
filter=userName%20eq%20%22alex.lee%40example.com%22
GET {SCIM_BASE_URL}/Users?filter=externalId%20eq%20%22idp-user-123%22
GET {SCIM_BASE_URL}/Groups?filter=displayName%20eq%20%22Sales%22
GET {SCIM_BASE_URL}/Groups?filter=externalId%20eq%20%22idp-group-456%22


```

Always URL-encode filters.

Avoid unsupported operators such as `ne`, `co`, `sw`, `ew`, and `not` unless the customer environment is explicitly verified to support them.

4.5 Sorting

The service code has internal sort handling for some list responses, but `ServiceProviderConfig` advertises sorting as unsupported. Middleware must not depend on server-side sorting.

4.6 Bulk, Password, and ETag

Do not use:

- SCIM Bulk
- SCIM password changes
- ETags or `If-Match` concurrency control

They are not supported by the Creatio SCIM app.

4.7 Request Size

Maximum request body size is 10 MB. Keep group membership PATCH batches below this limit. For large groups, split membership changes into multiple PATCH requests.

5. Discovery Endpoints

Discovery requests are not required before every provisioning action, but the middleware should support them for setup validation and health checks.

5.1 Service Provider Configuration

Use case:

- Validate that the token works.

- Discover high-level supported capabilities.

Request:

```
GET {SCIM_BASE_URL}/ServiceProviderConfig
Authorization: Bearer {ACCESS_TOKEN}
Accept: application/scim+json


```

Expected behavior:

- `200 OK`
- Returns SCIM service configuration.
- Current implementation advertises:
  - `patch.supported = true`
  - `filter.supported = true`
  - `bulk.supported = false`
  - `sort.supported = false`
  - `changePassword.supported = false`
  - `etag.supported = false`
  - OAuth 2.0 bearer authentication

Example response shape:

```
{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:ServiceProviderConfig"
  ],
  "patch": { "supported": true },
  "bulk": {
    "supported": false,
    "maxOperations": 0,
    "maxPayloadSize": 0
  },
  "filter": {
    "supported": true,
    "maxResults": 200
  },
  "sort": { "supported": false },
  "changePassword": { "supported": false },
  "etag": { "supported": false },
  "authenticationSchemes": [
    {
      "name": "OAuth 2.0 Bearer",
      "type": "oauth2",
      "primary": true
    }
  ]
}


```

Middleware requirement:

- Call this during setup and periodic health checks.
- Do not fail if extra fields are present.
- Do fail if authentication fails or if `patch.supported` or `filter.supported` is false in a customer environment where provisioning depends on PATCH and lookup filters.

5.2 Resource Types

Use case:

- Confirm that Creatio exposes User and Group resource types.

Request:

```
GET {SCIM_BASE_URL}/ResourceTypes
```

Expected behavior:

- `200 OK`
- Returns a SCIM `ListResponse`.
- Contains resources with IDs `User` and `Group`.

Specific resource type requests:

```
GET {SCIM_BASE_URL}/ResourceTypes/User
GET {SCIM_BASE_URL}/ResourceTypes/Group
```

Middleware requirement:

- Implement for diagnostics.
- The provisioning flow can continue without calling these on every run.

5.3 Schemas

Use case:

- Confirm schema metadata for User and Group.

Request:

```
GET {SCIM_BASE_URL}/Schemas
```

Expected behavior:

- `200 OK`
- Returns a SCIM `ListResponse`.
- Contains schemas for:
  - `urn:ietf:params:scim:schemas:core:2.0:User`
  - `urn:ietf:params:scim:schemas:core:2.0:Group`

Specific schema requests must URL-encode the schema URN:

```
GET {SCIM_BASE_URL}/Schemas/urn%3Aietf%3Aparams%3Ascim%3Aschemas%3Acore%3A2.0%3AUser
GET {SCIM_BASE_URL}/Schemas/urn%3Aietf%3Aparams%3Ascim%3Aschemas%3Acore%3A2.0%3AGroup

```

Middleware requirement:

- Implement for diagnostics and onboarding.
- Do not use schema discovery as a substitute for the concrete requirements in this document.

6. User Resource Model

Creatio supports these user attributes for provisioning:

SCIM attribute	Required	Creatio behavior
`schemas`	Recommended	Use `urn:ietf:params:scim:schemas:core:2.0:User`. If enterprise extension values are sent, also include urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.
`id`	Response only	Creatio SCIM User ID. Store it in middleware.
`externalId`	Strongly recommended	Stable IdP user identifier. Used for lookup and duplicate

		prevention. Cleared on user soft delete.
`userName`	Yes	Maps to `SysAdminUnit.Name`. Primary matching key.
`active`	Yes for middleware	Maps to `SysAdminUnit.Active` when materialized. Send `true` for active users.
`emails[type eq "work"].value`	Strongly recommended	Maps to `SysAdminUnit.Email` and `Contact.Email`. Used as a secondary matching key.
`displayName`	Recommended	Maps to `Contact.Name`.
`name.formatted`	Optional	Used for `Contact.Name` when `displayName` is absent.
`name.givenName`	Optional	Maps to `Contact.GivenName`.
`name.familyName`	Optional	Maps to `Contact.Surname`.
`name.middleName`	Optional	Maps to `Contact.MiddleName`.
`title`	Optional	Maps to `Contact.JobTitle`.

`phoneNumbers[type eq "work"].value`	Optional	Maps to `SysAdminUnit.Phone` and `Contact.Phone`.
`phoneNumbers[type eq "mobile"].value`	Optional	Maps to `Contact.MobilePhone`.
`addresses[type eq "work"].formatted`	Optional	Maps to `Contact.Address`.
`preferredLanguage`	Optional	Maps to `SysAdminUnit.SysCultureId` and `Contact.LanguageId` if the language/culture exists in Creatio.
`urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.employeeNumber`	Optional	Stored in SCIM projection custom attributes. Returned under Enterprise User extension in GET responses. Not materialized to standard `SysAdminUnit` / `Contact` fields.
`urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.department`	Optional	Stored in SCIM projection custom attributes. Returned under Enterprise User extension in GET responses. Not materialized to standard

		`SysAdminUnit` / `Contact` fields.
`urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.costCenter`	Optional	Stored in SCIM projection custom attributes. Returned under Enterprise User extension in GET responses. Not materialized to standard `SysAdminUnit` / `Contact` fields.
`urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.division`	Optional	Stored in SCIM projection custom attributes. Returned under Enterprise User extension in GET responses. Not materialized to standard `SysAdminUnit` / `Contact` fields.
`urn:ietf:params:scim:schemas:extension:enterprise:2.0:User.organization`	Optional	Stored in SCIM projection custom attributes. Returned under Enterprise User extension in GET responses. Not materialized to standard `SysAdminUnit` / `Contact` fields.
`urn:ietf:params:scim:schemas:extension:enterprise:2`	Optional	Stored in SCIM projection custom attributes as manager

`.0:User.manager.value`		identifier. Returned as `manager.value` inside Enterprise User extension in GET responses.
`groups`	Usually response; PATCH supported	Represents group membership. Middleware should prefer group-centric membership via `PATCH /Groups/{id}`.
`meta`	Response only	Contains `resourceType`, `created`, `lastModified`, and `location`.

Avoid relying on unsupported or no-op user fields:

- `nickName` is accepted by DTO but is not a meaningful Creatio provisioning field.
- `name.honorificPrefix` and `name.honorificSuffix` can be stored in the projection response but do not map to standard Creatio Contact fields in the current mapper.
- Enterprise extension fields are supported for projection persistence and response round-trip, but they are not mapped to standard Creatio `SysAdminUnit` or `Contact` fields.
- Manager metadata is normalized to `manager.value`. Incoming `manager.displayName` and `manager.$ref` are accepted for compatibility but are not persisted.
- Address subfields other than `formatted` are not materialized to Creatio.

7. User Use Cases

7.1 Lookup User by UserName

Use this before creating or updating a user when the middleware does not already have the Creatio SCIM User ID.

Request:

```
GET {SCIM_BASE_URL}/Users?
filter=userName%20eq%20%22alex.lee%40example.com%22&startIndex=1&count=
```

```
100
Authorization: Bearer {ACCESS_TOKEN}
Accept: application/scim+json


```

Expected response:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:ListResponse"
  ],
  "totalResults": 1,
  "startIndex": 1,
  "itemsPerPage": 1,
  "Resources": [
    {
      "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:User"
      ],
      "id": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
      "externalId": "idp-user-123",
      "userName": "alex.lee@example.com",
      "active": true,
      "meta": {
        "resourceType": "User"
      }
    }
  ]
}


```

Middleware behavior:

- If `totalResults = 0`, create the user.
- If `totalResults = 1`, store or refresh the Creatio SCIM User ID and update by `PATCH`.
- If `totalResults > 1`, stop and report a data integrity problem. `userName` is expected to be unique inside a SCIM profile.

7.2 Lookup User by External ID

Use this if the middleware stores stable IdP IDs and wants a stronger identity lookup.

Request:

```
GET {SCIM_BASE_URL}/Users?filter=externalId%20eq%20%22idp-user-123%22


```

Middleware behavior:

- If found, use the returned `id`.
- If not found, fall back to `userName` lookup before creating.
- Do not assume `externalId` remains present after a user is deleted from Creatio SCIM projection. The current delete flow clears it.

7.3 Get User by ID

Use this when the middleware already stores the Creatio SCIM User ID.

Request:

```
GET {SCIM_BASE_URL}/Users/8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11


```

Expected behavior:

- `200 OK` with the user resource if found.
- `404 notFound` if the ID is not a valid Creatio SCIM User ID or does not exist.
- If the projection user was soft-deleted, the current implementation can return it by ID with `active=false` and no `externalId`. List queries exclude soft-deleted users.

Middleware behavior:

- Use this for reconciliation and diagnostics.
- If `404`, remove the stale mapping and attempt lookup by `externalId` or `userName`.

7.4 Create User

Use this when lookup confirms that the user does not already exist in Creatio SCIM.

Request:

```
POST {SCIM_BASE_URL}/Users
Authorization: Bearer {ACCESS_TOKEN}
Accept: application/scim+json
Content-Type: application/scim+json

{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:User"
  ],
  "externalId": "idp-user-123",
  "userName": "alex.lee@example.com",
  "active": true,
  "displayName": "Alex Lee",
  "name": {
    "formatted": "Alex Lee",
    "givenName": "Alex",
    "familyName": "Lee"
  },
  "emails": [
    {
      "value": "alex.lee@example.com",
      "type": "work",
      "primary": true
    }
  ],
  "title": "Account Executive",
  "phoneNumbers": [
    {
      "value": "+1 555 0100",
```

```
      "type": "work",
      "primary": true
    },
    {
      "value": "+1 555 0199",
      "type": "mobile"
    }
  ],
  "preferredLanguage": "en-US"
}


```

Optional Enterprise User extension payload (commonly sent by Entra) can be included on create:

```
{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:User",
    "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
  ],
  ...
  "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User": {
    "employeeNumber": "EMP-2001",
    "department": "Platform",
    "costCenter": "CC-01",
    "division": "Revenue",
    "organization": "Example",
    "manager": {
      "value": "58ac0edf-36f3-4e37-8e4d-6f5f3f4b05d8"
    }
  }
}
```

Expected success:

- `201 Created`
- `Location` header points to the created SCIM user.
- Response body contains the Creatio SCIM User `id`.

Creatio behavior:

- Creatio stores the user in the SCIM projection layer immediately.
- Creatio provisions the user to a real Creatio `SysAdminUnit` only when the user is eligible:
  - the user belongs to at least one SCIM group already provisioned to a Creatio role, or
  - the SCIM profile is configured to auto-provision users without group membership to the default role, or
  - an administrator manually provisions the groupless user from SCIM Administration.
- If the user is not yet eligible, the SCIM request can still return success because the projection layer was updated.
- Creatio matching uses:
  - SCIM `userName` to Creatio `SysAdminUnit.Name`

    - SCIM work email to Creatio `SysAdminUnit.Email`
- If `userName` and work email identify different existing Creatio users, provisioning can fail with `409 uniqueness` or produce a failed-user record for admin review.

Middleware behavior:

- Store the returned `id`.
- Do not assume the user can log in to Creatio immediately after `201`. Role/group provisioning and Creatio admin review may still be needed.
- Do not retry a successful `201` as another create. Use `PATCH` for subsequent changes.
- Enterprise extension values are persisted when this call creates a new projection user.
- If this create request is internally resolved as an update of an existing projection user (same userName/email) or soft-deleted user reactivation, enterprise extension values are not reliably persisted in the current implementation. Use `PATCH` enterprise paths after identity resolution to enforce enterprise metadata.

7.5 Full Replace User

Endpoint:

```
PUT {SCIM_BASE_URL}/Users/{id}


```

The current repository implements this endpoint, although Entra-style provisioning normally uses `PATCH`.

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:User"
  ],
  "externalId": "idp-user-123",
  "userName": "alex.lee@example.com",
  "active": true,
  "displayName": "Alex Lee",
  "name": {
    "formatted": "Alex Lee",
    "givenName": "Alex",
    "familyName": "Lee"
  },
  "emails": [
    {
      "value": "alex.lee@example.com",
      "type": "work",
      "primary": true
    }
  ]
}


```

Expected behavior:

- `200 OK` with updated user.
- `404 notFound` if user does not exist.
- `400 invalidSyntax` if required fields are missing.
- `409 uniqueness` if `userName` or `externalId` conflicts with another projection user.

Middleware requirement:

- Prefer `PATCH` for routine changes.
- Use `PUT` only if the middleware intentionally sends the full desired state every time.
- When using `PUT`, include all fields that must remain in Creatio. Missing optional fields can be cleared or omitted from the projection.

7.6 Patch User Attributes

Use this for routine user updates.

Endpoint:

```
PATCH {SCIM_BASE_URL}/Users/{id}


```

The request body must contain `schemas` and `Operations`. Use capital `Operations` for compatibility with the current DTO.

Example: update display name and title.

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "displayName",
      "value": "Alex Morgan Lee"
    },
    {
      "op": "replace",
      "path": "title",
      "value": "Senior Account Executive"
    }
  ]
}


```

Supported user PATCH operations:

- `add`

- `replace`
- `remove`

Supported user PATCH paths:

- `userName`
- `displayName`
- `active`
- `externalId`
- `title`
- `preferredLanguage`
- `name`
- `name.givenName`
- `name.familyName`
- `name.middleName`
- `name.formatted`
- `name.honorificPrefix`
- `name.honorificSuffix`
- `emails`
- `emails[type eq "work"].value`
- `phoneNumbers`
- `phoneNumbers[type eq "work"].value`
- `phoneNumbers[type eq "mobile"].value`
- `addresses`
- `addresses[type eq "work"].formatted`
- `groups`
- `groups[value eq "{groupId}"]`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:employeeNumber`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:department`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:costCenter`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:division`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:organization`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager`

- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager.value`

Accepted no-op compatibility paths:

- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager.displayName`
- `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager.$ref` (or `.ref`)
- Other `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager.*` sub-attributes are accepted and ignored.
- `addresses[type eq "work"]` subfields such as `streetAddress`, `locality`, `region`, `postalCode`, `country` are accepted for compatibility but are not materialized. Use `formatted`.

Enterprise manager behavior:

- `manager` can be sent as a SCIM complex object or as a scalar string value.
- `manager.value` is persisted as the normalized manager identifier.
- `manager.displayName` and `manager.$ref` are accepted but not persisted.
- `manager` arrays are invalid and return `400 invalidValue`.

Example: update work email.

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "emails[type eq \"work\"].value",
      "value": "alex.lee2@example.com"
    }
  ]
}
```

Example: update a full name object.

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "name",
      "value": {
```

```
        "formatted": "Alex M. Lee",
        "givenName": "Alex",
        "middleName": "Morgan",
        "familyName": "Lee"
      }
    }
  ]
}


```

Example: remove title.

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "remove",
      "path": "title"
    }
  ]
}


```

Example: patch enterprise extension attributes.

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path":
"urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:employeeNu
mber",
      "value": "EMP-3001"
    },
    {
      "op": "replace",
      "path":
"urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:department
",
      "value": "Revenue Operations"
    },
    {
      "op": "replace",
      "path":
"urn:ietf:params:scim:schemas:extension:enterprise:2.0:User:manager",
      "value": {
        "value": "d4f8c4f7-1534-4d8f-bfd9-249ad7223d66"
      }
    }
  ]
}
```

Creatio behavior:

- Creatio merges the PATCH into the current projection user.

- Unmentioned attributes are preserved.
- If the user is already materialized to Creatio, supported fields are applied to `SysAdminUnit` and `Contact`.
- Enterprise extension values are persisted in SCIM projection custom attributes and returned in SCIM user responses.
- The enterprise schema URN is included in user response `schemas` only when at least one enterprise extension value is present.
- If materialization fails after the projection update, Creatio can still return success and record the materialization problem in provisioning logs or failed-user areas. The projection layer remains the source of truth for SCIM GET responses.

Middleware behavior:

- Use `replace` for attribute changes.
- Use `remove` only for attributes that should really be cleared.
- Do not send unsupported paths. Unsupported paths return `400 invalidValue`.

7.7 Deactivate User

This is the preferred Entra-compatible user deprovisioning behavior.

Request:

```
PATCH {SCIM_BASE_URL}/Users/{id}


```

Body:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "active",
      "value": false
    }
  ]
}


```

Expected behavior:

- `200 OK` with user resource.
- Projection user becomes inactive.
- If already materialized, Creatio sets `SysAdminUnit.Active = false`.

- Creatio removes the user from elevated roles and preserves the configured default role where applicable.
- Creatio logs user deactivation events.

Middleware behavior:

- Use this when the IdP disables, suspends, or removes the user from provisioning scope but does not require the SCIM resource to disappear from Creatio.
- Also remove the user from group memberships as appropriate using `PATCH /Groups/{groupId}` remove operations.

7.8 Reactivate User

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "active",
      "value": true
    }
  ]
}
```

Expected behavior:

- `200 OK`.
- Projection user becomes active.
- If materialized, Creatio sets `SysAdminUnit.Active = true`.
- Role access depends on current group memberships and Creatio group-to-role mappings. Add or restore group memberships after reactivation if the user should regain role access.

7.9 Delete User

Endpoint:

```
DELETE {SCIM_BASE_URL}/Users/{id}
```

The current repository implements this endpoint. The public app guide may not list it for all package versions, so confirm the deployed customer package before relying on it.

Expected behavior:

- `204 No Content` on success.
- `404 notFound` if the projection user does not exist.
- Idempotent if the projection user was already soft-deleted.

Creatio behavior:

- Soft-deletes the SCIM projection user.
- Sets projection `active=false`.
- Clears projection `externalId`.
- If the user was materialized to Creatio:
  - sets `SysAdminUnit.Active=false`
  - removes elevated roles except the configured default role
  - deletes SCIM provisioning bindings and membership tracking
- Does not physically delete the Creatio user/contact.

Middleware behavior:

- Prefer `PATCH active=false` to mimic Entra deactivation behavior.
- Use `DELETE /Users/{id}` only if the customer wants the SCIM resource removed from normal list responses and the customer package is verified to support it.

7.10 Modify User Groups from the User Endpoint

The current implementation supports user-centric group membership PATCH operations. This is useful, but group-centric membership via `PATCH /Groups/{id}` is recommended for Entra-like behavior.

Add a user to groups:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "add",
      "path": "groups",
      "value": [
        {
          "value": "4a5c393f-f674-478d-94e7-561c5af62533"
        }
      ]
    }
  ]
}
```

Remove a user from a specific group:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "remove",
      "path": "groups[value eq \"4a5c393f-f674-478d-94e7-
561c5af62533\"]"
    }
  ]
}
```

Replace all group memberships for a user:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "groups",
      "value": [
        {
          "value": "4a5c393f-f674-478d-94e7-561c5af62533"
        },
        {
          "value": "93a50830-8bdb-48c0-b434-b7922e84d42c"
        }
      ]
    }
  ]
}
```

Middleware behavior:

- Use Creatio SCIM Group IDs in `groups.value`.
- Prefer group endpoint membership operations unless there is a strong middleware reason to update membership from the user side.

8. Group Resource Model

Creatio supports these group attributes:

SCIM attribute	Required	Creatio behavior
`schemas`	Recommended	Use `urn:ietf:params:scim:schemas:core:2.0:Group`.

`id`	Response only	Creatio SCIM Group ID. Store it in middleware.
`externalId`	Strongly recommended	Stable IdP group identifier. Used for lookup and duplicate prevention.
`displayName`	Yes	Stored in the SCIM projection group. Used for Creatio role suggestion/mapping.
`members`	Optional	List of Creatio SCIM User IDs in the group. Used for role assignment when group is mapped to a Creatio role.
`meta`	Response only	Contains resource metadata.

Nested groups are not supported. If a member has `"type": "Group"`, the current implementation skips it.

9. Group Use Cases

Important behavior note:

- `GET /Groups` is not profile-specific in the current implementation and can return groups across profiles.

9.1 Lookup Group by Display Name

Request:

```
GET {SCIM_BASE_URL}/Groups?
filter=displayName%20eq%20%22Sales%22&excludedAttributes=members

```

Middleware behavior:

- If `totalResults = 0`, create the group.
- If `totalResults = 1`, store or refresh the returned Creatio SCIM Group ID.
- If `totalResults > 1`, stop and report a data integrity problem.

9.2 Lookup Group by External ID

Request:

```
GET {SCIM_BASE_URL}/Groups?filter=externalId%20eq%20%22idp-group-456%22&excludedAttributes=members


```

Middleware behavior:

- Prefer this if the IdP has stable group IDs.
- Fall back to displayName lookup only if externalId lookup fails.

9.3 Get Group by ID

Request:

```
GET {SCIM_BASE_URL}/Groups/4a5c393f-f674-478d-94e7-561c5af62533


```

Expected behavior:

- `200 OK` if the group exists and is not deleted.
- `404 notFound` if the group does not exist or was deleted.

9.4 Create Group

Request:

```
POST {SCIM_BASE_URL}/Groups
Authorization: Bearer {ACCESS_TOKEN}
Accept: application/scim+json
Content-Type: application/scim+json

{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:Group"
  ],
  "externalId": "idp-group-456",
  "displayName": "Sales",
  "members": [
    {
      "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
      "type": "User"
    }
  ]
}


```

Expected success:

- `201 Created`
- `Location` header points to the created group.
- Response includes the Creatio SCIM Group `id`.

Creatio behavior:

- Creates a SCIM projection group.
- Creates or updates group membership records in the projection layer.
- Places the group into the SCIM Administration flow for role mapping:
  - If exactly one Creatio role matches the group name, Creatio can suggest it.
  - If no role matches, Creatio can prefill a new role name.
  - If multiple roles match, administrator selection is required.
- Creatio does not automatically create or map a Creatio role solely because a group was received from SCIM. Administrator review/provisioning is expected.
- Once the group is provisioned to a Creatio role, membership changes trigger Creatio role assignment for users.

Middleware behavior:

- Store the returned group `id`.
- Notify the Creatio administrator that newly received groups may require provisioning in the SCIM Administration page.
- Do not expect role access to change until the group is mapped/provisioned in Creatio.

9.5 Full Replace Group

Endpoint:

```
PUT {SCIM_BASE_URL}/Groups/{id}


```

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:Group"
  ],
  "externalId": "idp-group-456",
  "displayName": "Sales Team",
  "members": [
    {
      "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
      "type": "User"
    },
    {
```

```
      "value": "60412b11-8f9d-46b1-bf80-4c18f548a8d3",
      "type": "User"
    }
  ]
}
```

Expected behavior:

- `200 OK` with updated group.
- Required field: `displayName`.
- Replaces all group members with the supplied `members` list.

Creatio behavior:

- Updates the SCIM projection group.
- Recalculates role suggestion if the group is awaiting provisioning.
- If already provisioned to a Creatio role, membership changes update Creatio role assignment for affected users.

Middleware behavior:

- Use `PUT` only when sending the complete desired group state.
- For incremental membership changes, prefer `PATCH /Groups/{id}`.

9.6 Patch Group Display Name

Use this when the IdP group is renamed.

Request:

```
PATCH {SCIM_BASE_URL}/Groups/{id}
```

Body:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "displayName",
      "value": "Revenue Operations"
    }
  ]
}
```

Expected behavior:

- `200 OK` with updated group.

Creatio behavior:

- Updates the projection group name returned by SCIM GET.
- If the group is awaiting provisioning, role suggestions are recalculated.
- If the group is already mapped to a Creatio role, the mapping remains linked to the same Creatio role. Creatio does not automatically rename the Creatio role.

9.7 Add Members to Group

Request:

```
PATCH {SCIM_BASE_URL}/Groups/{groupId}


```

Body:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "add",
      "path": "members",
      "value": [
        {
          "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
          "type": "User"
        },
        {
          "value": "60412b11-8f9d-46b1-bf80-4c18f548a8d3",
          "type": "User"
        }
      ]
    }
  ]
}


```

Expected behavior:

- `200 OK`.
- Operation is idempotent for duplicate members.
- User IDs must exist in Creatio SCIM projection. Non-existent user IDs produce `400 invalidValue`.
- Member entries with invalid GUID format in `members.value` can be skipped by the current implementation instead of failing the entire request.

Creatio behavior:

- Adds membership in the projection layer.
- If the group has already been provisioned to a Creatio role, Creatio assigns that role to materialized users.
- If a user is not yet materialized but is now eligible because of group membership, Creatio can attempt to provision the user.

Middleware behavior:

- Create or lookup users first.
- Use only returned Creatio SCIM User IDs.
- For large groups, batch membership additions to keep payload below 10 MB.

9.8 Remove One Member from Group

Preferred Entra-compatible removal form:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "remove",
      "path": "members[value eq \"8e0e7a4f-9f2d-4f8e-98a9-
6adcad8e0a11\"]"
    }
  ]
}
```

Also accepted by the current implementation:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "remove",
      "path": "members",
      "value": [
        {
          "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11"
        }
      ]
    }
  ]
}
```

Expected behavior:

- `200 OK`.

- Removing a non-member is safe and results in no membership for that user in that group.

Creatio behavior:

- Removes membership in the projection layer.
- If the group is mapped to a Creatio role, removes the user from that role.
- If the user loses their last provisioned role, Creatio assigns the configured default role or root fallback where applicable.
- User active status is not changed by group membership removal.

9.9 Remove All Members from Group

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "remove",
      "path": "members"
    }
  ]
}
```

Expected behavior:

- `200 OK`.
- Replaces membership with an empty set.

Middleware behavior:

- Use carefully. This removes every member from the SCIM group and can remove users from the mapped Creatio role.

9.10 Replace All Members in Group

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "members",
      "value": [
        {
          "value": "8e0e7a4f-9f2d-4f8e-98a9-6adcad8e0a11",
          "type": "User"
        },
```

```
        {
          "value": "60412b11-8f9d-46b1-bf80-4c18f548a8d3",
          "type": "User"
        }
      ]
    }
  ]
}
```

Expected behavior:

- `200 OK`.
- Existing members not listed are removed.
- Listed users are added or kept.

Middleware behavior:

- Use only when the middleware has the full desired member list.
- For incremental provisioning, use `add` and filtered `remove`.

9.11 Replace One Member in Group

The current implementation supports filtered member replacement.

Request:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:PatchOp"
  ],
  "Operations": [
    {
      "op": "replace",
      "path": "members[value eq \"8e0e7a4f-9f2d-4f8e-98a9-
6adcad8e0a11\"]",
      "value": [
        {
          "value": "60412b11-8f9d-46b1-bf80-4c18f548a8d3",
          "type": "User"
        }
      ]
    }
  ]
}
```

Middleware behavior:

- Usually not needed for Entra-like flows.
- Prefer explicit remove old member then add new member.

9.12 Delete Group

Request:

```
DELETE {SCIM_BASE_URL}/Groups/{id}


```

Expected behavior:

- `204 No Content`.
- `404 notFound` if group does not exist or does not belong to the SCIM profile.

Creatio behavior:

- Soft-deletes the SCIM projection group.
- Deletes projection membership rows.
- Deletes unprovisioned/provisioned group tracking rows.
- If the group was mapped to a Creatio role:
  - removes SCIM-managed users from that role
  - assigns default role/root fallback if a user would otherwise have no role
- Does not delete the Creatio role.
- Does not deactivate users.
- Does not automatically remove licenses outside normal role membership effects.

Middleware behavior:

- Use when an IdP group is deleted or permanently removed from provisioning scope.
- Do not expect the corresponding Creatio role to disappear.

10. Recommended End-to-End Flows

10.1 Initial Full Synchronization

1. Acquire OAuth token.
2. Call `GET /ServiceProviderConfig` to validate SCIM availability.
3. For each IdP user:
   - Lookup by `externalId`.
   - If not found, lookup by `userName`.
   - If not found, `POST /Users`.
   - If found, `PATCH /Users/{id}` to align attributes.
   - Store the Creatio SCIM User `id`.
4. For each IdP group:

- Lookup by `externalId`.
- If not found, lookup by `displayName`.
- If not found, `POST /Groups`.
- If found and renamed, `PATCH /Groups/{id}` displayName.
- Store the Creatio SCIM Group `id`.

5. For each group membership:
   - Ensure every member has a Creatio SCIM User `id`.
   - Use `PATCH /Groups/{id}` with `add` operations.
6. Ask the Creatio administrator to review Groups awaiting provisioning in the SCIM Administration page and map/provision groups to Creatio roles.
7. Re-run membership synchronization after groups are provisioned if needed, or rely on Creatio to materialize eligible users after group mapping.

10.2 Incremental User Create

1. Lookup user by `externalId`.
2. If not found, lookup by `userName`.
3. If still not found, `POST /Users`.
4. Store returned `id`.
5. Add the user to groups using `PATCH /Groups/{groupId}`.

10.3 Incremental User Attribute Update

1. Resolve Creatio SCIM User `id` from middleware mapping, or lookup.
2. Send `PATCH /Users/{id}` with changed attributes.
3. On `404`, clear the mapping and run lookup by `externalId` and `userName`.
4. On `409`, stop and ask the Creatio administrator to resolve the userName/email conflict.

10.4 User Deactivation

1. Resolve Creatio SCIM User `id`.
2. Send `PATCH /Users/{id}` with `active=false`.
3. Remove from groups if the IdP says the user no longer belongs to them.
4. Do not use direct role removal APIs.

10.5 User Reactivation

1. Resolve Creatio SCIM User `id`.
2. Send `PATCH /Users/{id}` with `active=true`.
3. Add back group memberships.

4. Confirm expected access after Creatio group-to-role mappings apply.

10.6 Group Create and Provisioning

1. Lookup group by `externalId`, then `displayName`.
2. If missing, `POST /Groups`.
3. Store returned group `id`.
4. Notify the Creatio administrator to provision the received group to a Creatio role.
5. Send or re-send membership operations after users and groups exist.

10.7 Group Rename

1. Resolve Creatio SCIM Group `id`.
2. Send `PATCH /Groups/{id}` with `displayName`.
3. Do not expect Creatio role names to be renamed automatically.

10.8 Group Membership Add/Remove

1. Resolve group `id`.
2. Resolve user `id`.
3. Use `PATCH /Groups/{id}`:
   - `add` for new members.
   - `remove` with `members[value eq "{userId}"]` for removed members.
4. For large syncs, batch operations.

10.9 Group Deletion

1. Resolve Creatio SCIM Group `id`.
2. Send `DELETE /Groups/{id}`.
3. Remove the mapping from middleware storage after `204`.
4. Do not expect the Creatio role to be deleted.

11. Error Handling

Creatio SCIM errors use the SCIM error response shape:

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:Error"
  ],
  "status": "400",
  "scimType": "invalidValue",
  "detail": "Human-readable error message"
}
```

Handle these statuses:

Status	Meaning	Middleware action
`200`	Read/update success	Continue. Store returned resource state.
`201`	Create success	Store returned `id` and `Location`.
`204`	Delete success	Remove local mapping if deleting.
`400 invalidSyntax`	Malformed request, missing required field, invalid GUID, invalid JSON	Do not retry unchanged. Fix middleware payload.
`400 invalidValue`	Unsupported PATCH operation/path, invalid member, invalid value	Do not retry unchanged. Fix payload or data.
`400 invalidFilter`	Unsupported or malformed filter	Fix lookup filter.
`401 invalidCredentials`	Missing/expired/invalid token	Refresh token and retry once. If it repeats, stop.
`404 notFound`	Resource not found	Clear stale mapping; lookup by `externalId` / `userName` or create if appropriate.
`409 uniqueness`	Duplicate/conflicting `userName`, `externalId`, or userName/email mapping	Stop. Requires data cleanup or admin resolution.

`413`	Payload too large	Split the request into smaller batches.
`500`	Server error	Retry with backoff if operation is idempotent; otherwise stop and escalate with request correlation logs.

12. Middleware Data Store Requirements

The middleware should persist at least:

```
IdP user immutable id
IdP user current userName
Creatio SCIM User id
Last successful user sync timestamp
IdP group immutable id
IdP group current displayName
Creatio SCIM Group id
Last successful group sync timestamp
Membership sync state or change cursor


```

The middleware must be able to recover if its mapping table is incomplete:

1. Lookup by `externalId`.
2. Fallback lookup by `userName` for users or `displayName` for groups.
3. Store the returned Creatio SCIM `id`.

13. Attribute and Behavior Notes for Creatio Administrators

User provisioning:

- A SCIM user can exist in the projection layer before a Creatio user is created or updated.
- A real Creatio user is created or linked only when provisioning eligibility is met.
- Users without group membership require the SCIM profile default-role behavior or manual action.

Group provisioning:

- Incoming SCIM groups are flat.
- Creatio administrators map groups to existing roles or create new roles from the SCIM Administration page.

- One IdP group maps to one Creatio role within the same SCIM profile.
- One Creatio role cannot be mapped to multiple IdP groups within the same SCIM profile.

Group deletion:

- Deletes SCIM group projection and membership state.
- Does not delete Creatio roles.
- Does not deactivate users.

User deletion/deactivation:

- `PATCH active=false` deactivates the user when materialized.
- `DELETE /Users/{id}` soft-deletes the SCIM projection user and deactivates the linked Creatio user if present.

14. Acceptance Tests

The customer middleware is considered compatible when these tests pass in a non-production Creatio environment.

14.1 Authentication and Discovery

1. Token request succeeds.
2. `GET /ServiceProviderConfig` returns `200`.
3. `GET /ResourceTypes` returns User and Group.
4. `GET /Schemas` returns User and Group schemas.

14.2 User CRUD and Lifecycle

1. Lookup unknown user by `externalId` returns `totalResults=0`.
2. `POST /Users` creates the user and returns `201` with an `id`.
3. `GET /Users/{id}` returns the user.
4. `GET /Users?filter=userName eq "..."` returns exactly one user.
5. `PATCH /Users/{id}` updates `displayName`, `name`, work email, title, work phone, mobile phone, and preferred language.
6. `PATCH /Users/{id}` with `active=false` deactivates the projection user and, if materialized, the Creatio user.
7. `PATCH /Users/{id}` with `active=true` reactivates the projection user and, if materialized, the Creatio user.
8. Duplicate `externalId` or conflicting `userName` produces `409`.

14.3 Group CRUD and Membership

1. Lookup unknown group by `externalId` returns `totalResults=0`.
2. `POST /Groups` creates the group and returns `201` with an `id`.
3. `GET /Groups/{id}` returns the group.
4. `PATCH /Groups/{id}` updates `displayName`.
5. `PATCH /Groups/{id}` adds a known user member.
6. `GET /Groups/{id}` returns that member.
7. `PATCH /Groups/{id}` removes that member with `members[value eq "{userId}"]`.
8. `PATCH /Groups/{id}` replace members sets the exact desired member list.
9. `DELETE /Groups/{id}` returns `204`; later `GET /Groups/{id}` returns `404`.

14.4 Creatio-Specific Administration

1. Newly received groups appear in SCIM Administration for provisioning.
2. Administrator can map a group to an existing role or create a new role.
3. After group provisioning, adding a user to the group assigns the mapped Creatio role.
4. Removing the user from the group removes the mapped Creatio role and applies default-role behavior if needed.

15. Version and Compatibility Notes

This guide reflects the current repository implementation reviewed on 2026-05-19.

Known compatibility boundary: this implementation was validated with Microsoft Entra behavior, but this repository does not contain raw Entra provisioning traffic logs or a formal compatibility matrix for other IdPs. The requirements below are derived from the current Creatio SCIM code, the public Creatio SCIM app instructions, and Entra-compatible SCIM request shapes. A customer middleware must pass the acceptance tests in a Creatio sandbox before production use.

Validate the installed customer package before relying on endpoints that are implemented in the repository but may not appear in all public-facing app instructions, especially:

- `PUT /Users/{id}`
- `DELETE /Users/{id}`

For Entra-like behavior, the safest minimum set is:

- `GET /ServiceProviderConfig`
- `GET /Users`

- `GET /Users/{id}`
- `POST /Users`
- `PATCH /Users/{id}`
- `GET /Groups`
- `GET /Groups/{id}`
- `POST /Groups`
- `PUT /Groups/{id}` only for full replacement
- `PATCH /Groups/{id}`
- `DELETE /Groups/{id}`
- `GET /Schemas`
- `GET /Schemas/{id}`
- `GET /ResourceTypes`
- `GET /ResourceTypes/{id}`

16. References

- Creatio Marketplace: SCIM Integration - SCIM Integration | Creatio Marketplace
- Creatio SCIM Integration Instructions PDF - https://marketplace.creatio.com/sites/marketplace/files/app-guide/SCIM Integration Instructions.pdf
- Microsoft Entra SCIM provisioning guidance - Tutorial - Develop a SCIM endpoint for user provisioning to apps from Microsoft Entra ID - Microsoft Entra ID
- SCIM 2.0 protocol RFC 7644 - RFC 7644: System for Cross-domain Identity Management: Protocol
- SCIM 2.0 core schema RFC 7643 - RFC 7643: System for Cross-domain Identity Management: Core Schema