API Manuals

- Advertiser & Invoicing Companies, Brands, and User Mapping

Advertiser & Invoicing Companies, Brands, and User Mapping

This guide explains how to create **advertiser companies**, **invoicing companies**, and **brands** in Adhese through the Campaign API, and how to **connect (map) users** to those companies and brands.

All endpoints live under the Adhese API and are versioned with a `/v1` prefix. The server base path is `/api`, so a full path looks like `/api/v1/media-partners` and `/v1/user-mapping`.

Key concepts & terminology

In Adhese, an advertiser company and an invoicing company are **the same underlying entity — a *media partner*** — distinguished by the **roles** assigned to it. A single media partner can hold one or more roles at the same time.

You want to create…	What it is in the API	How
An **advertiser company**	A media partner with the `ADVERTISER` role	`POST /v1/media-partners` with `"roles": ["ADVERTISER"]`
An **invoicing company**	A media partner with the `INVOICE` role	`POST /v1/media-partners` with `"roles": ["INVOICE"]`
A company that is **both**	A media partner with both roles	`POST /v1/media-partners` with `"roles": ["ADVERTISER", "INVOICE"]`
A **brand**	A *media brand* that belongs to a media partner	`POST /v1/media-partners/{mediaPartnerId}/brands`
A **user ↔ company/brand link**	A *user mapping*	`POST /v1/user-mapping`

Available roles: `ADVERTISER`, `INVOICE`

> **Note.** The read-only endpoints `GET /v1/advertiser-companies` and `GET /v1/brands` expose the same entities from the campaign-booking perspective (used when

> creating or editing a campaign). Their `id` values are the media-partner and media-brand ids you create below.

Authentication & headers

Every request is authenticated with a **Bearer JWT** and must carry the Keycloak auth header.

Header	Required	Value	Notes
`Authorization`	Yes	`Bearer <jwt>`	JWT access token.
`Use-Keycloak-Auth`	Yes	`true`	Required on all `/v1` endpoints.
`Content-Type`	For `POST` / `DELETE` with a body	`application/json`	—

Example request line and headers:

```
POST /api/v1/media-partners
Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...
Use-Keycloak-Auth: true
Content-Type: application/json
```

Create advertiser & invoicing companies

Both are created with the **same endpoint**; the `roles` array is what makes a company an advertiser, an invoicing party, or both.

Create a company

```
POST /v1/media-partners
```

Creates a media partner (advertiser company, invoicing company, or both, depending on `roles`).

Request body — `CreateMediaPartnerRequest`

Field	Type	Required	Description

`name`	string	Yes	Display name of the company.
`roles`	array of enum	Yes	Any of `ADVERTISER`, `INVOICE`
`externalKey`	string	No	Your own external reference key.
`subsystemExternalIds`	object (string → string)	No	External ids per subsystem.

Create an advertiser company

```
{
  "name": "Acme Beverages",
  "roles": ["ADVERTISER"],
  "externalKey": "acme-bev",
  "subsystemExternalIds": {
    "crm": "CRM-10432"
  }
}
```

Create an invoicing company

```
{
  "name": "Acme Beverages Billing BV",
  "roles": ["INVOICE"],
  "externalKey": "acme-bev-billing"
}
```

Create a company that is both advertiser and invoicing party

```
{
  "name": "Acme Beverages",
  "roles": ["ADVERTISER", "INVOICE"]
}
```

Responses

Status	Meaning
`201`	Media partner created.
`400`	Invalid input (e.g. empty `name` or unknown role).
`401` / `403`	Not authenticated / not allowed.
`500`	Unexpected failure.

> The `201` response will return a body including the generated `id` and all information known about the media partner.

List companies

```
GET /v1/media-partners?limit=50&offset=0
```

Parameter	In	Required	Type	Description
`limit`	query	Yes	integer	Max number of results to return.
`offset`	query	Yes	integer	Number of results to skip.
`search`	query	No	string	URL-encoded, case-insensitive match on name.
`roles`	query	No	array	Filter on the role(s) the media partner has, e.g. `roles=ADVERTISER`.
`includeInactive`	query	No	boolean	Include deactivated media partners.

Response — array of `MediaPartner Schema`

```
[
  {
    "id": 4021,
    "name": "Acme Beverages",
    "roles": ["ADVERTISER"],
    "externalKey": "acme-bev",
    "subsystemExternalIds": { "crm": "CRM-10432" },
    "active": true
  }
]
```

Get a single company

```
GET /v1/media-partners/{mediaPartnerId}
```

Parameter	In	Required	Type	Description
`mediaPartnerId`	path	Yes	integer	The id of the media partner.

Returns a single `MediaPartner Schema` (same shape as above).

Create brands

A brand is a **media brand** that belongs to a media partner (typically the advertiser company).

Create a brand

```
POST /v1/media-partners/{mediaPartnerId}/brands
```

Parameter	In	Required	Type	Description
`mediaPartnerId`	path	Yes	integer	The media partner (company) the brand belongs to.

Request body — `CreateMediaBrandRequest`

Field	Type	Required	Description
`name`	string	Yes	Brand name.
`externalKey`	string	No	Your own external reference key.
`subsystemExternalIds`	object (string → string)	No	External ids per subsystem.

```
{
  "name": "Acme Cola",
  "externalKey": "acme-cola",
  "subsystemExternalIds": {}
}
```

Responses

Status	Meaning
`201`	Media brand created.
`400`	Invalid input.

Status	Meaning
`401` / `403`	Not authenticated / not allowed.
`404`	Media partner not found.

> As with companies, `201` will return a body including the generated `id` and all information known about the brand.

List a company's brands

```
GET /v1/media-partners/{mediaPartnerId}/brands?limit=50&offset=0
```

Parameter	In	Required	Type	Description
`mediaPartnerId`	path	Yes	integer	The media partner.
`limit`	query	Yes	integer	Max number of results.
`offset`	query	Yes	integer	Results to skip.
`includeInactive`	query	No	boolean	Include deactivated brands.
`search`	query	No	string	URL-encoded, case-insensitive match on name.

Response — array of `MediaBrand Schema`

```
[
  {
    "id": 8801,
    "name": "Acme Cola",
    "externalKey": "acme-cola",
    "subsystemExternalIds": {},
    "active": true
  }
]
```

Get a single brand

```
GET /v1/media-partners/{mediaPartnerId}/brands/{mediaBrandId}
```

Returns a single `MediaBrand Schema`.

Connect users to media partners

A **user mapping** grants a user access to a company/brand combination. Each mapping entry requires an `advertiserCompanyId` **and** a `brandId`; `invoiceCompanyId` is optional.

Create a user mapping

```
POST /v1/user-mapping
```

Request body — `CreateUserMappingRequest`

Field	Type	Required	Description
`user`	string (1–255)	Yes	User identifier (e.g. email or username).
`mappings`	array of `Mapping Schema`	Yes	At least one mapping entry.

Mapping Schema

Field	Type	Required	Description
`advertiserCompanyId`	integer (≥ 1)	Yes	The advertiser company to grant access to.
`invoiceCompanyId`	integer	No	Restrict the mapping to a specific invoicing company.
`brandId`	integer (≥ 1)	Yes	The brand to grant access to.

```
{
  "user": "jane.doe@partner.example",
  "mappings": [
    {
      "advertiserCompanyId": 4021,
      "invoiceCompanyId": 4022,
      "brandId": 8801
    }
  ]
}
```

Responses

Status	Meaning
`201`	User mapping created.
`400`	Invalid input (e.g. missing `advertiserCompanyId` or `brandId`).
`401` / `403`	Not authenticated / not allowed.

List users with mappings

```
GET /v1/user-mapping?limit=50&offset=0
```

Parameter	In	Required	Type	Description
`limit`	query	No	integer	Max results (≤ 100).
`offset`	query	No	integer	Results to skip.
`search`	query	No	string	Loose match on user identifier.
`advertiserCompanyId`	query	No	integer	Only users mapped to this advertiser company.
`invoiceCompanyId`	query	No	integer	Only users mapped to this invoice company.
`brandId`	query	No	integer	Only users mapped to this brand.

The response includes a `Record-Count` header with the total number of matches.

Response — array of `UserMapping Schema`

```
[
  { "user": "jane.doe@partner.example", "mappingCount": 3 }
]
```

List a single user's mappings

```
GET /v1/user-mapping/{user}
```

Parameter	In	Required	Type	Description
`user`	path	Yes	string	The user identifier.

Parameter	In	Required	Type	Description
`limit` / `offset`	query	No	integer	Pagination.
`advertiserCompanyId`	query	No	integer	Filter by advertiser company.
`invoiceCompanyId`	query	No	integer	Filter by invoice company.
`brandId`	query	No	integer	Filter by brand.

Response — array of `Mapping Schema`

```
[
  {
    "advertiserCompanyId": 4021,
    "invoiceCompanyId": 4022,
    "brandId": 8801
  }
]
```

Delete a user mapping

```
DELETE /v1/user-mapping
```

Request body — `DeleteUserMappingRequest`

Field	Type	Required	Description
`user`	string (1–255)	Yes	The user identifier.
`advertiserCompanyId`	integer (≥ 1)	—	Advertiser company of the mapping to remove.
`invoiceCompanyId`	integer	—	Invoice company of the mapping to remove.
`brandId`	integer (≥ 1)	—	Brand of the mapping to remove.

```
{
  "user": "jane.doe@partner.example",
  "advertiserCompanyId": 4021,
  "invoiceCompanyId": 4022,
  "brandId": 8801
}
```

Responds `200` when the mapping is deleted.

End-to-end walkthrough

Create an advertiser company, an invoicing company, and a brand, then give a user access to them.

1. Create the advertiser company

```
POST /v1/media-partners

{
  "name": "Acme Beverages",
  "roles": ["ADVERTISER"],
  "externalKey": "acme-bev"
}
```

The response will give you its id → assume `4021`.

2. Create the invoicing company

```
POST /v1/media-partners

{
  "name": "Acme Beverages Billing BV",
  "roles": ["INVOICE"],
  "externalKey": "acme-bev-billing"
}
```

The response will give you its id → assume `4022`.

3. Create a brand under the advertiser company

```
POST /v1/media-partners/4021/brands

{
  "name": "Acme Cola",
  "externalKey": "acme-cola"
}
```

The response will give you its id → assume `8801`.

4. Map the user to the company + brand

```
POST /v1/user-mapping

{
```

```
  "user": "jane.doe@partner.example",
  "mappings": [
    { "advertiserCompanyId": 4021, "invoiceCompanyId": 4022, "brandId": 8801 }
  ]
}
```

5. Verify the mapping

```
GET /v1/user-mapping/jane.doe@partner.example
```

```
[
  { "advertiserCompanyId": 4021, "invoiceCompanyId": 4022, "brandId": 8801 }
]
```

Error handling

Errors are returned as an RFC 7807 `ProblemDetail` object.

```
{
  "type": "about:blank",
  "title": "Bad Request",
  "status": 400,
  "detail": "roles must not be empty",
  "instance": "/api/v1/media-partners"
}
```

Status	Meaning
`400`	Bad Request — invalid input or parameters.
`401`	Unauthorized — authentication required or invalid token.
`403`	Forbidden — authenticated but not allowed to access this resource.
`404`	Not Found — resource not found.
`422`	Unprocessable Entity — the request was understood but could not be processed.
`500`	Internal Server Error — unexpected failure.

Schema reference

CreateMediaPartnerRequest · MediaPartner Schema

```
{
  "id": 4021,
  "name": "string",
  "roles": ["ADVERTISER", "INVOICE", "INTERMEDIARY", "MEDIA"],
  "externalKey": "string",
  "subsystemExternalIds": { "subsystem": "externalId" },
  "active": true
}
```

CreateMediaBrandRequest · MediaBrand Schema

```
{
  "id": 8801,
  "name": "string",
  "externalKey": "string",
  "subsystemExternalIds": { "subsystem": "externalId" },
  "active": true
}
```

AdvertiserCompany Schema

```
{
  "id": 4021,
  "name": "string",
  "active": true,
  "externalId": "string",
  "subsystemExternalIds": { "subsystem": "externalId" }
}
```

Brand Schema

```
{ "id": 8801, "name": "string" }
```

CreateUserMappingRequest Schema

```
{
  "user": "string",
  "mappings": [
    { "advertiserCompanyId": 4021, "invoiceCompanyId": 4022, "brandId": 8801 }
  ]
}
```

UserMapping Schema

```
{ "user": "string", "mappingCount": 3 }
```