# integeri HTTP API v1 — Integration mit Make.com

**Version:** Stand Repository (API v1)
**Zielgruppe:** Externe Integratoren (Make.com / Integromat)
**Basis-URL:** `https://ihre-domain.de` (Beispiel lokal: `http://localhost`)

---

## 1. Überblick

Dieses Dokument beschreibt ein **Make.com-Szenario** aus Sicht eines externen Anwenders der integeri-API:

1. **Anmeldung** → OAuth2 Access Token (`client_id` + `client_secret`)
2. **Standort ermitteln** (Filter) → `location_id`, Name der Baustelle
3. **Team anlegen** am Standort (Teamname = Name der Baustelle) → `team_id`, `recurring_course_id`
4. **Videoinhalt** in den Team-Unterweisungskurs (`recurring_course_id`) einfügen → `content_node_id`

**Optional (Erweiterung):** Abgeschlossene Kurse eines Benutzers listen und **Zertifikat-PDF** über `download_url` herunterladen (Module 6–8).

Die API legt beim Team automatisch einen **wiederkehrenden Unterweisungskurs** an (`recurring_course_id`). Das Video wird diesem Kurs zugeordnet, indem Sie `team_id` mitsenden — ohne manuelle Kurs-IDs.

Weitere Dokumentation:

| Ressource | URL / Pfad |
|---|---|
| HTML-API-Doku (DE) | `{BASE_URL}/docs/api/v1` |
| HTML-API-Doku (EN) | `{BASE_URL}/docs/api/v1/en` |
| PHP Mini-SDK (ZIP) | `{BASE_URL}/docs/api/integeri-sdk-php.zip` |
| **Make.com-Anleitung (PDF)** | `{BASE_URL}/docs/api/integeri-api-make-com.pdf` |
| Make.com-Anleitung (Word) | `{BASE_URL}/docs/api/integeri-api-make-com.docx` |
| Kurzdoku (Repo) | `docs/api/v1.md` |

---

## 2. Voraussetzungen

Sie benötigen von integeri:

| Was | Beschreibung |
|---|---|
| **client_id** | Öffentliche Kennung Ihres API-Clients |
| **client_secret** | Geheimes Passwort (nur bei Erstellung einmal im Klartext — sicher in Make speichern) |
| **Basis-URL** | Ihre integeri-Instanz, z. B. `https://ihre-domain.de` |

Speichern Sie `client_id` und `client_secret` in Make.com z. B. im **Data Store** oder als verschlüsselte Szenario-Variablen.

Der mit dem API-Client verknüpfte Benutzer muss in integeri u. a. Standorte lesen, Teams anlegen und Inhalte erstellen dürfen (`location.viewAny`, `team.create`, `content.create`). Details zu Rollen

klären Sie mit Ihrem integeri-Ansprechpartner.

---

## 3. Szenario-Architektur in Make.com

### 3.1 Modul-Übersicht

| Nr. | Modultyp | Zweck | Ausgabe (wichtig) |
|---|---|---|---|
| 1 | HTTP | Token holen | `access_token`, `expires_in` |
| 2 | HTTP | Standorte listen | `data[]` |
| 3 | Iterator **oder** Set variable | Standort wählen | `location_id`, `team_name` |
| 4 | HTTP | Team anlegen | `id` (team_id), `recurring_course_id` |
| 5 | HTTP | Video im Team-Kurs | `content_node_id` |
| 6 | HTTP | Abgeschlossene Kurse | `data[]` mit `download_url` |
| 7 | Iterator / Filter | Eintrag mit PDF | `download_url`, `assignment_id` |
| 8 | HTTP | PDF laden | Binärdatei (PDF) |

### 3.2 Ablauf (Sequenz)

**Hauptszenario (Team + Video):**

```
[Trigger] → [1 Token] → [2 Locations] → [3 location_id + team_name] → [4 Team] → [5 Video]
```

**Erweiterung Zertifikat-PDF** (nach Modul 1, unabhängig vom Video-Flow):

```
[1 Token] → [6 Certificated courses] → [7 download_url wählen] → [8 PDF GET] → [Speichern / E-Mail / …]
```

**Fachliche Reihenfolge:** Zuerst Team an der Baustelle (Standort), danach das Unterweisungsvideo in den Kurs dieses Teams (`recurring_course_id`).

---

## 4. Modul 1 — Access Token (Login)

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{BASE_URL}}/oauth/token` |
| Method | `POST` |
| Headers | `Content-Type: application/json` |
| Body type | Raw / JSON |

**Request Body:**

```
{
  "grant_type": "client_credentials",
  "client_id": "{{IHR_CLIENT_ID}}",
  "client_secret": "{{IHR_CLIENT_SECRET}}",
  "scope": "*"
}
```

**Erfolg (HTTP 200):** JSON mit u. a.:

| Feld | Mapping in Make (Beispiel) |
|---|---|
| `access_token` | `{{1.access_token}}` |
| `expires_in` | `{{1.expires_in}}` (Sekunden) |
| `token_type` | meist `Bearer` |

**Empfehlung:** Szenario-Variable `integeri_token` = `{{1.access_token}}`.
Alle folgenden Module: Header `Authorization: Bearer {{integeri_token}}`

**Fehler:** 401 bei falschem Secret.

**Optional — Modul 1b:** `GET {{BASE_URL}}/api/v1/auth/me` mit Bearer-Header → prüft den Service-User.

---

## 5. Modul 2 — Standorte listen (mit Filter)

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{BASE_URL}}/api/v1/locations` |
| Method | `GET` |
| Headers | `Authorization: Bearer {{integeri_token}}`, `Accept: application/json` |
| Query String | siehe unten |

### 5.1 Query-Parameter

| Parameter | Pflicht | Beispiel | Beschreibung |
|---|---|---|---|
| `state` | nein | `active` | Nur aktive Standorte |
| `search` | nein | `Hamburg` | Suche im Standortnamen (max. 100 Zeichen) |
| `per_page` | nein | 25 | 1–100 |
| `page` | nein | 1 | Paginierung |

**Beispiel-URL:**

```
{{BASE_URL}}/api/v1/locations?state=active&search=Baustelle&per_page=25&page=1
```

**Erfolg (HTTP 200):** `data[]` mit u. a. `id`, `state`, `contact.name`.

---

## 6. Modul 3 — Standort und Teamname ermitteln

### Variante A — Feste Werte vom Trigger

**Modul:** `Tools` → **Set multiple variables**

| Variable | Wert |
|---|---|
| `location_id` | `{{trigger.location_id}}` |
| `team_name` | `{{trigger.team_name}}` |

### Variante B — Erster Standort aus der Liste

**Modul:** `Flow control` → **Iterator** — Array: `{{2.data}}`

Pro Bundle (ggf. **Stop after one bundle**):

| Variable | Wert |
|---|---|
| `location_id` | `{{3.id}}` |
| `team_name` | `{{3.contact.name}}` |

Der **Teamname** entspricht in der Regel dem **Namen der Baustelle** (`contact.name` des Standorts).

### Variante C — Filter nach Standortname

Nach dem Iterator: **Filter** — z. B. `{{3.contact.name}}` enthält `Baustelle`.

### Variante D — Router

Route 1: `location_id` vom Trigger gesetzt → direkt zu Modul 4.
Route 2: sonst Iterator wie Variante B.

---

## 7. Modul 4 — Team anlegen

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{BASE_URL}}/api/v1/teams` |
| Method | `POST` |
| Headers | `Authorization: Bearer {{integeri_token}}`, `Accept: application/json`, `Content-Type: application/json` |

**Request Body:**

```
{
  "name": "{{team_name}}",
  "location_id": {{location_id}},
  "description": "Angelegt via Make.com"
}
```

**Hinweise zu Make-Mappings:**

- `name` = Name der Baustelle (z. B. `{{team_name}}` aus Modul 3).
- `location_id` ist eine **Zahl** (ohne Anführungszeichen in JSON).
- `description` optional.

**Erfolg (HTTP 201):**

```
{
  "data": {
    "id": 12,
    "type": "team",
    "location_id": 42,
    "recurring_course_id": 88,
    "contact": { "name": "Baustelle Nord", ... }
  }
}
```

**Variablen für Modul 5:**

| Variable | Mapping |
|---|---|
| `team_id` | `{{4.data.id}}` |
| `recurring_course_id` | `{{4.data.recurring_course_id}}` |

| Variable | Mapping |
|---|---|

*(Modulnummer `4` an Ihr Szenario anpassen.)*

`recurring_course_id` ist der **wiederkehrende Unterweisungskurs** des Teams. Er wird bei der Team-Anlage von integeri bereitgestellt.

---

## 8. Modul 5 — Videoinhalt in den Team-Unterweisungskurs

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{BASE_URL}}/api/v1/contentvideos/create` |
| Method | `POST` |
| Headers | wie Modul 4 |

**Request Body:**

```
{
  "prompter_text": "{{IHR_TELEPROMPTER_TEXT}}",
  "team_id": {{team_id}}
}
```

- **`prompter_text`:** Ihr Teleprompter- / Sicherheitstext (Pflichtfeld).
- **`team_id`:** ID aus Modul 4 (`{{4.data.id}}`). Die API ordnet das Video dem Unterweisungskurs dieses Teams (`recurring_course_id`) zu — Sie müssen **keine Kurs-ID** (`course_id`) kennen oder mitschicken.

**Erfolg (HTTP 201):**

```
{
  "message": "OK",
  "data": {
    "content_node_video_id": 1001,
    "content_node_id": 2002,
    "course_node_id": 3003,
    "course_id": 88,
    "estimated_video_length": 16
  }
}
```

Hier ist `data.course_id` der **Team-Unterweisungskurs** (entspricht `recurring_course_id`).

---

## 9. Erweiterung — Zertifikat-PDF herunterladen

Voraussetzungen: `courseAssignment.viewAny` und `UserPolicy::view` auf den Ziel-User.
Der Benutzer muss mindestens einen abgeschlossenen Kurs mit `download_available = true` haben.

### 9.1 Modul 6 — Abgeschlossene Kurse listen

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{BASE_URL}}/api/v1/users/{{user_id}}/certificated-courses` |

| Einstellung | Wert |
|---|---|
| Method | `GET` |
| Headers | `Authorization: Bearer {{integeri_token}}`, `Accept: application/json` |

`user_id` kommt vom Trigger oder einem vorherigen Schritt (integeri-Benutzer-ID).

**Erfolg (HTTP 200):** `data[]` mit u. a.:

| Feld | Bedeutung |
|---|---|
| `id` | Zuweisungs-ID (`assignment`) |
| `state_label` | Lokalisiertes Ergebnis (z. B. „Bestanden“) |
| `download_available` | `true`, wenn die Web-UI einen Download anbietet |
| `download_url` | API-PDF-URL oder signierte Upload-URL; `null` wenn kein Download |
| `course.id` | Kurs-ID (für Kontrolle) |

**Beispiel `download_url` (API-PDF):**

```
https://ihre-domain.de/api/v1/users/42/99/course/15-de.pdf
```

Bei **hochgeladenen** Zertifikatsdateien ist `download_url` eine **signierte Storage-URL** — dort **keinen** Bearer-Header mitsenden.

### 9.2 Modul 7 — Eintrag mit Download wählen

**Modul:** `Flow control` → **Iterator** — Array: `{{6.data}}`

**Filter** (pro Bundle):

```
{{7.download_available}} = true
AND {{7.download_url}} is not empty
```

Optional **Stop after one bundle**, wenn nur das erste Zertifikat benötigt wird.

**Variablen:**

| Variable | Mapping |
|---|---|
| `download_url` | `{{7.download_url}}` |
| `assignment_id` | `{{7.id}}` |
| `course_title` | `{{7.course.title}}` |

**Kein Download** bei u. a. `short_type = c_pd`, Gruppen-Bediener-Schulung (`is_group_operator_training`), ausstehendem BKF (`exchange_pending`).

### 9.3 Modul 8 — PDF herunterladen

**Modul:** `HTTP` → **Make a request**

| Einstellung | Wert |
|---|---|
| URL | `{{download_url}}` |
| Method | `GET` |
| Parse response | **No** (Rohdaten / Binär) |

**Header (nur bei API-PDF-URL):**

Wenn `download_url` mit `{{BASE_URL}}/api/` beginnt:

| Header | Wert |
|---|---|
| `Authorization` | `Bearer {{integeri_token}}` |

Bei signierten Upload-URLs **keinen** Authorization-Header setzen.

**Erfolg (HTTP 200):** Antwortkörper ist die **PDF-Datei** (`Content-Type: application/pdf`).

**Folgemodule in Make:** z. B. **Google Drive** → Upload a file, **Dropbox** → Upload, **E-Mail** → Anhang, oder **Webhooks** → Base64.

**Beispiel Router für Header:**

| Route | Bedingung | Modul 8 |
|---|---|---|
| A | `download_url` enthält `/api/v1/users/` | GET mit Bearer |
| B | sonst | GET ohne Bearer (signierte URL) |

### 9.4 cURL-Referenz (Zertifikat-PDF)

```
USER_ID=42
CERT=$(curl -s "$BASE_URL/api/v1/users/$USER_ID/certificated-courses" \
  -H "Authorization: Bearer $TOKEN" -H "Accept: application/json")
DOWNLOAD_URL=$(echo "$CERT" | jq -r '.data[] | select(.download_url != null) |
.download_url' | head -1)

# API-PDF-Route → Bearer mitsenden:
curl -s -o zertifikat.pdf "$DOWNLOAD_URL" -H "Authorization: Bearer $TOKEN"
file zertifikat.pdf
```

---

## 10. Fehlerbehandlung in Make.com

### 10.1 HTTP-Error-Handler

Nach jedem HTTP-Modul: **Error handler**

| Status | Aktion |
|---|---|
| 401 | Modul 1 wiederholen (Token neu) |
| 403 | API-Zugang / Berechtigungen prüfen |
| 422 | Body `errors` ausgeben (Validierung) |

### 10.2 Typische 422-Felder

| Feld | Ursache |
|---|---|
| `location_id` | Standort unbekannt oder andere Firma |
| `name` | Teamname fehlt oder ungültig |
| `team_id` | Team unbekannt oder ohne Unterweisungskurs |
| `prompter_text` | Leer oder zu lang |

---

## 11. Vollständige Mapping-Tabelle

| Make-Variable | Quelle | Verwendung in |
|---|---|---|
| `integeri_token` | Modul 1 `access_token` | Bearer-Header |

| Make-Variable | Quelle | Verwendung in |
|---|---|---|
| `location_id` | Trigger / Modul 3 `id` | POST /teams |
| `team_name` | Modul 3 `contact.name` | POST /teams → `name` |
| `team_id` | Modul 4 `data.id` | POST /contentvideos/create → `team_id` |
| `recurring_course_id` | Modul 4 `data.recurring_course_id` | Kontrolle / Folgeprozesse |
| `content_node_id` | Modul 5 `data.content_node_id` | Folgeprozesse |
| `user_id` | Trigger / vorheriger Schritt | GET /certificated-courses |
| `download_url` | Modul 7 `download_url` | GET PDF (Modul 8) |
| `assignment_id` | Modul 7 `id` | Logging / Dateiname |

---

## 12. cURL-Referenz (Test außerhalb von Make)

```
export BASE_URL="https://ihre-domain.de"
export CLIENT_ID="..."
export CLIENT_SECRET="..."

TOKEN=$(curl -s -X POST "$BASE_URL/oauth/token" \
  -H "Content-Type: application/json" \
  -d "
{\"grant_type\":\"client_credentials\",\"client_id\":\"$CLIENT_ID\",\"client_secret\":\"$CL
\
  | jq -r .access_token)

curl -s "$BASE_URL/api/v1/locations?state=active&per_page=5" \
  -H "Authorization: Bearer $TOKEN" -H "Accept: application/json" | jq .

TEAM=$(curl -s -X POST "$BASE_URL/api/v1/teams" \
  -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"name":"Baustelle Test","location_id":1,"description":"cURL"}')
echo "$TEAM" | jq .
TEAM_ID=$(echo "$TEAM" | jq -r .data.id)

curl -s -X POST "$BASE_URL/api/v1/contentvideos/create" \
  -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d "{\"prompter_text\":\"Test aus cURL\",\"team_id\":$TEAM_ID}" | jq .
```

---

## 13. Checkliste vor Go-Live

- [ ] `BASE_URL` in Make auf Produktiv-Domain gesetzt
- [ ] `client_id` und `client_secret` sicher hinterlegt (nicht in Logs)
- [ ] Testlauf: Team-Antwort mit `recurring_course_id` > 0
- [ ] Testlauf: Video-Antwort mit `content_node_id` und passender `course_id`
- [ ] Optional: Testlauf Zertifikat — `certificated-courses` liefert `download_url`, PDF-Download liefert `%PDF`
- [ ] Fehler-Route bei 422 mit Benachrichtigung

---

*integeri GmbH — Externe API-Dokumentation für Make.com. Technische Details:* `/docs/api/v1`.