

Interface Control Document (ICD)

Employee Data Feed — External API (EDF-EXT)

Document Title	Employee Data Feed External API — Interface Control Document
System	webapi-employeeDataFeed-ext
API Version	v1 (URL-path versioned; <code>Asp.Versioning</code>)
Interface Style	REST / JSON over HTTPS
Gateway	Azure API Management (APIM)
Status	Baseline, revision 3 (September 2026) — §3.1 token endpoint and valid scopes, §10 Data Provider Onboarding added; onboarding contact eOPFOperations@opm.gov
Classification	Public — for external integration partners (contains no secrets or internal identifiers)

1. Introduction

This Interface Control Document (ICD) defines the external interface exposed by the **Employee Data Feed (EDF) External API**. It describes the transport, security, versioning, endpoints, request/response data contracts, and error semantics that provider systems must observe when integrating with the EDF. The API accepts employee records (insert or update) into the eOPF employee data store and provides authentication and error-catalog diagnostics.

1.1 Scope

This ICD covers the public, versioned HTTP surface (`/v1/*`) served behind Azure API Management. Internal implementation details (database schema and internal processing pipelines) are out of scope except where they affect the observable interface contract.

1.2 Table of Contents

2. [Transport & Conventions](#)
3. [Security & Authentication](#)
4. [Versioning](#)
5. [Endpoint Catalog](#)
6. [Data Contracts](#)
7. [Error Handling](#)
8. [Correlation & Observability](#)
9. [Provenance Logging](#)
10. [Data Provider Onboarding for API Access](#)

2. Transport & Conventions

Attribute	Value
Protocol	HTTPS (TLS 1.2+) only
Content-Type	<code>application/json</code>
Produces	<code>application/json</code>
Character Set	UTF-8
Base Path	<code>https://<apim-gateway-host>/<api-suffix>/v1</code>
Date Format	Payload dates are strings <code>MM/dd/yyyy</code> ; timestamps in diagnostics are ISO 8601 / RFC 3339 UTC

3. Security & Authentication

The API is protected by two layers: the subscription key is enforced at the APIM gateway; the JWT bearer token is enforced at the gateway and re-validated by the API:

Layer	Mechanism	Where
-------	-----------	-------

APIM Subscription Key	Header <code>Ocp-Apim-Subscription-Key: <key></code> on every call	Provisioned per subscriber by the API administrator
OAuth2 / Entra ID JWT Bearer	Header <code>Authorization: Bearer <token></code> via the client-credentials flow (machine-to-machine)	Token from the OPM tenant endpoint <code>https://login.microsoftonline.com/3ce07c0b-30c5-4bcd-ad92-238975b649af/oauth2/v2.0/token</code> with the EDF scope for the tier being called (§3.1)

3.1 Token endpoint and valid scopes

Field	Value
Flow	<code>client_credentials</code> (service-to-service; no user interaction)
Token endpoint	<code>https://login.microsoftonline.com/3ce07c0b-30c5-4bcd-ad92-238975b649af/oauth2/v2.0/token</code> (OPM tenant)
Issuer validated by the gateway	<code>https://sts.windows.net/3ce07c0b-30c5-4bcd-ad92-238975b649af/</code>
<code>client_id</code> / <code>client_secret</code>	Issued to your system at onboarding (§10); store the secret in a secrets manager
Token lifetime	~60 minutes — cache the token and refresh shortly before expiry; do not mint one per call

Tier	Valid EDF scope (request exactly this value)
Non-production (TST / UAT)	<code>api://4d4da1e8-fc9b-41ff-8318-b3f7ec5b4ca4/.default</code>
Production (PRD)	<code>api://b3f239c0-39ac-450c-a96e-7e121f26c8ba/.default</code>

Scopes are feed-specific. Each eOPF data feed (IDF, EDF, FDF) has its own API audience, and non-production and production differ again. A token requested with another feed's scope, or a non-production token sent to production, is rejected at the gateway with `401`. The scope the EDF gateway expects is also returned by `GET /v1/Auth/config`.

3.2 Token request example (non-production)

Replace the placeholders with the client ID and client secret issued to your system; for production use your production credentials with the production scope above.

```
$body = @{
    client_id      = '<your-client-id>'
    client_secret  = '<your-client-secret>'      # from your secrets manager;
    never commit it
    scope          = 'api://4d4da1e8-fc9b-41ff-8318-b3f7ec5b4ca4/.default'  #
    EDF non-production
    grant_type     = 'client_credentials'
}
$token = (Invoke-RestMethod -Method Post -Body $body `
    -Uri 'https://login.microsoftonline.com/3ce07c0b-30c5-4bcd-ad92-
238975b649af/oauth2/v2.0/token').access_token
```

```
curl -s -X POST 'https://login.microsoftonline.com/3ce07c0b-30c5-4bcd-ad92-
238975b649af/oauth2/v2.0/token' \
    -d 'client_id=<your-client-id>' -d 'client_secret=<your-client-secret>' \
    -d 'scope=api://4d4da1e8-fc9b-41ff-8318-b3f7ec5b4ca4/.default' -d
'grant_type=client_credentials'
```

Authorization roles. The `EDF.OPS` application role (OPM-internal) grants access to all provider/subagency data and is required for the `queryEmployee` diagnostic endpoint. Per-provider subagency authorizations are recorded during onboarding and will be enforced as provider-specific access control is enabled.

New integrators should first call the anonymous `GET /v1/Auth/config` to discover authentication requirements and `GET /v1/Auth/me` to confirm their credentials are recognized.

4. Versioning

The API uses **URL-path versioning**. All endpoints are exposed under `/v{major}` (currently `/v1` ; future versions will be published as `/v2` , `/v3` , ...), and the version segment is **required** — unversioned paths return 404 everywhere, at the APIM gateway and against the backend alike. Supported versions are advertised in the `api-supported-versions` response header. (Internally, APIM's version set consumes the version segment while matching; each

version's Web service URL carries its matching suffix — `/v1` for `v1` — so the forwarded request is re-rooted at the versioned path for the versioned-only backend.)

5. Endpoint Catalog

Method	Path	Auth	Description
POST	<code>/v1/Employee/saveEmployee</code>	Bearer + Key	Validate and save (insert/update) an employee record
GET	<code>/v1/Employee/queryEmployee</code>	Bearer + Key + EDF.OPS	Diagnostic lookup of a persisted employee snapshot and recent load-detail audit rows
GET	<code>/v1/Auth/me</code>	Bearer + Key	Return caller identity and access state (401 without a valid token)
GET	<code>/v1/Auth/config</code>	Key (no Bearer)	Return authentication configuration and onboarding instructions
GET	<code>/v1/Employee/Documentation</code>	Key (no Bearer)	Endpoint documentation summary
GET	<code>/v1/errors</code>	Key (no Bearer)	Return the catalog of possible status codes and validation messages

5.1 POST `/v1/Employee/saveEmployee`

Performs an **insert** when the SSN is new and an **update** when it already exists. Returns `400` with validation messages when the payload is invalid or a referenced Agency/Subagency or Personnel Office Identifier does not exist.

Request body: `EmployeeRequestData` (see §6.1). **Response:** `EmployeeResponseData` (see §6.2).

5.2 GET `/v1/Employee/queryEmployee` EDF.OPS only

Returns the persisted employee snapshot and recent load-detail audit rows for the supplied SSN. The returned SSN is masked. Intended for verifying `saveEmployee` submissions, not for

bulk data extraction.

Request: query string parameters per `EmployeeQueryRequest` (see §6.3), e.g. `GET /v1/Employee/queryEmployee?SSN=900-00-0001` .

6. Data Contracts

6.1 EmployeeRequestData (saveEmployee request)

All properties are JSON strings. Wire JSON is camelCase; request property-name binding is case-insensitive, so PascalCase keys (`LastName`) also bind. A **required** field fails validation when it is missing, empty, or below its minimum length; every field fails when it exceeds its maximum length. Validation failures return `400` with eOPF error code `1301` (FieldMissing) or `1302` (FieldInvalid) unless a more specific code is listed below. All failures for a request are reported together.

Field (JSON key)	Type	Required	Constraints / limits	Description
<code>guid</code>	string	No	Must parse as a GUID to be honored; otherwise the server generates a new identifier.	Optional client-supplied record identifier.
<code>fileName</code>	string	No	Sanitized to <code>[A-Za-z0-9_--]</code> ; truncated at 255 characters. Falls back to the request <code>X-Correlation-Id</code> , then to the generated GUID.	Optional load identifier for the submission's audit row.
<code>agencyCode</code>	string	Yes	Exactly 4 characters.	Agency code for the load grouping.
<code>recordNumber</code>	string	Yes	1–6 characters.	Provider record number.
<code>lastName</code>	string	Yes	1–35 characters.	Last name.
<code>firstName</code>	string	Yes	1–35 characters.	First name.
<code>middleName</code>	string	No	Up to 35 characters.	Middle name.

login	string	No	Up to 20 characters.	User login.
ssn	string	Yes	Format ###-##-#### with dashes (pattern ^\d{3}-\d{2}-\d{4}\$). Failures return code 911 (InvalidSsn).	Social Security Number — the create/update matching key. Never returned or logged in full (only the last 4 digits are stored in provenance).
employeeId	string	No	Up to 20 characters.	Provider employee identifier.
personnelOfficeId	string	Yes	4–10 characters; must exist in eOPF or the call fails with 400 / code 1402 (PersonnelOfficeNotFound).	Personnel Office Identifier.
agencySubAgencyCode	string	Yes	Exactly 4 characters; missing, invalid, or not-found values all return code 929 (InvalidAgencyCode) — a value that does not exist fails with 400 "Agency/Subagency Code does not exist."	Agency/Subagency code.
organizationalCode	string	Yes	1–22 characters.	Organizational code.
activityCode	string	Yes	1–15 characters. Agencies that do not use the code send a placeholder (the file-feed convention is 1).	Activity code — assigned by the agency to the organizational component in which the employee works.
superuser	string	Yes	Exactly 1 character (Y / N).	Superuser flag.
supervisor	string	Yes	Exactly 1 character (Y / N).	Supervisor flag.

address1	string	No	Up to 50 characters.	Address line 1.
address2	string	No	Up to 50 characters.	Address line 2.
city	string	No	Up to 50 characters.	City.
state	string	No	Up to 50 characters.	State.
zip	string	No	Up to 10 characters.	Zip code.
country	string	No	Up to 50 characters.	Country.
phone	string	No	Up to 50 characters.	Contact phone.
status	string	Yes	1–25 characters; e.g. <code>ACTIVE</code> , <code>TERMINATED</code> , <code>RETIRED</code> , <code>INACTIVE</code> .	Employment status.
statusDate	string	Conditional	<code>MM/dd/yyyy</code> , max 10 characters; required when status is not ACTIVE (case-insensitive). Auto-defaulted to today (UTC) when the status is terminated/retired/inactive and the date is omitted. Future dates are allowed.	Status effective date.
birthDate	string	Yes	<code>MM/dd/yyyy</code> , max 10 characters; must not be a future date (codes <code>1303</code> <code>InvalidDate</code> / <code>1304</code> <code>FutureDate</code>).	Date of birth.
email	string	No	Up to 255 characters; must be a parseable email address (code <code>1305</code> <code>InvalidEmail</code>).	Work email.
sourceIdentifiers	string	Yes	1–14 characters.	Source system identifier(s).

SSN format differs by endpoint and by feed. `saveEmployee` requires the **dashed** `###-##-####` format; dashes are optional only on the `queryEmployee` SSN query

parameter (§6.3). The IDF External API requires 9 digits with *no* dashes.

Example request:

```
{
  "agencyCode": "TR00",
  "recordNumber": "100001",
  "lastName": "Anderson",
  "firstName": "John",
  "middleName": "Q",
  "login": "janderson",
  "ssn": "111-22-0001",
  "employeeId": "EMP0001",
  "personnelOfficeId": "P0123456",
  "agencySubAgencyCode": "AG01",
  "organizationalCode": "ORG-001",
  "activityCode": "ACT-001",
  "superuser": "N",
  "supervisor": "Y",
  "address1": "123 Main St",
  "address2": "Suite 400",
  "city": "Washington",
  "state": "DC",
  "zip": "20001",
  "country": "USA",
  "phone": "202-555-0100",
  "status": "ACTIVE",
  "statusDate": "",
  "birthDate": "01/15/1985",
  "email": "john.anderson@example.com",
  "sourceIdentifiers": "HRSYS-01"
}
```

6.2 EmployeeResponseData (saveEmployee response)

Field	Type	Notes
Success	boolean	true when the record was saved.
StatusCode	integer	HTTP status code (also set on the response).
Messages	string[]	Validation or error messages (empty on success).

ErrorMessages	object[]	Coded errors: each entry pairs <code>errorMessage</code> with its stable numeric <code>eopfErrorCode</code> (documented by <code>GET /v1/errors</code>) for programmatic handling. Empty on success.
---------------	----------	---

```
{
  "success": true,
  "statusCode": 200,
  "messages": [],
  "errorMessages": []
}
```

6.3 EmployeeQueryRequest (queryEmployee query string parameters)

Parameter	Type	Notes
SSN	string	Required. Dashes optional.

7. Error Handling

Errors are returned with the appropriate HTTP status code and, for validation failures, a populated `Messages` array. The full catalog of possible status codes and sample messages can be retrieved at `GET /v1/errors` .

Status	Meaning
200 OK	Operation succeeded.
400 Bad Request	Validation failed, empty body, or a referenced lookup value (Agency/Subagency, Personnel Office) does not exist.
401 Unauthorized	Subscription key or bearer token missing/invalid.
403 Forbidden	Caller is authenticated but does not hold the required role (e.g. <code>EDF.OPS</code>).
500 Internal Server Error	Unexpected server error (correlated via <code>X-Correlation-Id</code>).

8. Correlation & Observability

Every request is assigned a correlation identifier. Callers may supply `X-Correlation-Id` ; if absent, the API generates one. The value is echoed back on the `X-Correlation-Id` response header and stamped into audit logs, enabling end-to-end tracing across the gateway, API, and persistence layers. When `FileName` is omitted on `saveEmployee` , the correlation id is used as the load identifier.

9. Provenance Logging

Each employee-record operation writes a best-effort provenance row to an internal audit log capturing correlation id, result, HTTP status, duration, endpoint, and a summary of the operation (record GUID, subagency code, and the last four SSN digits only — the full SSN is never stored). Provenance writes never fail the originating operation.

10. Data Provider Onboarding for API Access

Access to the EDF External API is requested from, and provisioned by, the OPM eOPF team at eOPFOperations@opm.gov. The steps are:

1. **Complete the Provider Intake Form** and submit it to **eOPFOperations@opm.gov**.
The form is a separate document, `EDF-Provider-Intake-Form.pdf` , distributed with this ICD (relative location `../Intake-Form/`). It collects your system details, the environments requested, the subagency codes you will submit for, roles, technical contacts, your secrets-delivery channel, and the public source IP addresses to allow-list. It contains no secrets.
2. **The OPM eOPF team provisions** your Microsoft Entra ID app registration, provider and subagency authorization, and APIM subscription (primary and secondary keys).
3. **The OPM eOPF team delivers your credentials** (client secret and subscription keys) through the secure channel you selected on the form — never by email.
4. **Verify connectivity to the API in TST:**
 - a. acquire a token and call `GET /v1/Auth/me` through the TST gateway (base URL supplied with your credentials) with the headers in §3;
 - b. an HTTP `200` response with your identity JSON confirms both credential layers end to end.

Providers integrate in TST first, promote to UAT for sign-off, and receive production access after UAT sign-off; production is provisioned separately (app registration, gateway allow-list, and APIM subscription) on request through the same mailbox. Send questions about

onboarding, credential rotation, or changes to your egress IP addresses to **eOPFOperations@opm.gov**.

Interface Control Document — Employee Data Feed External API (EDF-EXT). Generated from the versioned `/v1` HTTP surface. For access requests and onboarding see §10 (contact **eOPFOperations@opm.gov**); for authentication discovery start with `GET /v1/Auth/config`.