

Public API Migration Guide

Everything that changes when you move from the legacy NowCerts Public API to the Momentum AMP Public API — authentication, list endpoints and REST endpoints, with the full endpoint mapping.

Audience: integration developers and technical partners | **Covers:** the complete documented endpoint surface | **Issued:** 29 July 2026

1 Why move to the Momentum AMP Public API

The Momentum AMP Public API is the platform's current integration surface. Migrating gets you a simpler authentication flow, a JSON contract you can bind to strongly typed models without post-processing, and access to endpoint areas that only exist on the new API.

One call to log in

A single JSON request with a username and password. No `grant_type`, no registered application, and no client secret to provision, store or rotate.

Properly typed responses

Booleans are booleans, numbers are numbers, and absent values are `null`. The legacy login returned every custom field as a string, which every client had to parse by hand.

Leaner list payloads

The legacy insured and policy lists embedded whole child collections in every row. Those are now separate, purpose-built endpoints, so a list query returns a list instead of a nested object graph.

New endpoint areas

Policy change requests, carrier and MGA management, agency-level file management, certificate previews and bulk Zapier operations exist only on the Momentum AMP API.

Query collections without custom parameters

List endpoints accept standard OData options — `$filter`, `$select`, `$expand`, `$orderby`, `$top`, `$skip`, `$count` — so several legacy purpose-built "Find..." endpoints are replaced by a filter you control.

Consistent, machine-readable errors

Validation failures return a predictable envelope with a per-field error map, and malformed queries return `400` instead of a generic server error.

2 What changes, at a glance

Most of the surface carries over untouched: the great majority of REST and list endpoints are reachable at the same path, with the same verb, and return the same shape. The work is concentrated in three places — authentication (Part 1), a small set of list endpoints (Part 2), and a small set of REST endpoints (Part 3).

	Legacy NowCerts Public API	Momentum AMP Public API
Login request	Form-urlencoded with <code>grant_type</code> , <code>username</code> , <code>password</code> , <code>client_id</code> , <code>client_secret</code> , <code>scope</code>	JSON with just <code>Username</code> and <code>Password</code>

	Legacy NowCerts Public API	Momentum AMP Public API
Login response	OAuth-style <code>snake_case</code> ; all custom values are strings	<code>PascalCase</code> with real JSON types
Request header	<code>Authorization: Bearer <token></code>	<code>Authorization: Bearer <token></code> UNCHANGED
Refresh	Same URL as login, via <code>grant_type=refresh_token</code>	Its own route; send the expired token and the refresh token
API keys	Sent as the <code>Authorization</code> header on every request	Exchanged once for a token, which you then use as normal
List endpoint URLs	<code>/api/{ListName}</code>	<code>/api/{ListName}</code> UNCHANGED
List query options	<code>\$filter \$select \$expand \$orderby \$top \$skip \$count</code>	Identical UNCHANGED
List response casing	<code>camelCase</code> properties	<code>camelCase</code> properties UNCHANGED
Failure response	<code>403</code> with an <code>error</code> / <code>error_description</code> body	<code>401</code> with an empty body

TWO DIFFERENT CASING CONVENTIONS — BY DESIGN, AND UNCHANGED FROM THE LEGACY API

List (OData) endpoints return `camelCase` property names on both APIs, so your list-handling code needs no casing change. The authentication and REST endpoints return `PascalCase`. Only the authentication response changed casing, because the legacy login used OAuth's `snake_case`; the REST endpoints were already `PascalCase`.

PART 1

Authentication

3 Authentication endpoints

Legacy endpoint	Momentum AMP endpoint	Status	What to do
POST <code>/api/token</code> <code>grant_type=password</code>	POST <code>/api/token</code>	CHANGED	Same path, new request and response format. See §4.
POST <code>/api/token</code> <code>grant_type=refresh_token</code>	POST <code>/api/token/refresh</code>	MOVED	Point your refresh call at the new route and send both tokens. See §6.
—	POST <code>/api/token/exchange-api-key</code> <code>?apiKey=...</code>	NEW	Replaces sending an API key on every request. See §7.

4 Logging in

4.1 The request

BEFORE — LEGACY

- Form-urlencoded body

NOW — MOMENTUM AMP

- JSON body

- `grant_type=password` required
- `username`, `password`
- `client_id` and `client_secret`, in the body or as a Basic header
- Optional `scope`

- `Username` and `Password` — that is the whole model
- No `grant_type`
- No client registration, `client_id` or `client_secret`
- No `scope`

Before

POST /api/token
Content-Type: application/x-www-form-urlencoded

`grant_type=password&username=agent@agency.com&password=secret&client_id=myApp`

Now

POST /api/token
Content-Type: application/json

```
{
  "Username": "agent@agency.com",
  "Password": "secret"
}
```

4.2 The response

Before — note every custom value is a string

```
{
  "access_token": "...",
  "token_type": "bearer",
  "expires_in": 101952, // seconds remaining
  "refresh_token": "9f2c1e7a4b8d4f1e9c0a3b5d7e6f8a1b",
  "as:client_id": "myApp",
  "userName": "agent@agency.com",
  "userId": "...", "userDisplayName": "...",
  "agencyId": "...", "agencyName": "...",
  "contactId": "...",
  "userType": "Agent", // name of the type
  "parentAgencyId": "", // empty string when none
  "isAgencyOwner": "false", // the string "false"
  "email": "agent@agency.com",
  ".issued": "Mon, 29 Jul 2026 08:00:00 GMT",
  ".expires": "Tue, 30 Jul 2026 12:19:12 GMT"
}
```

Now

```
{
  "AccessToken": "...",
  "RefreshToken": "...",
  "TokenType": "JWT",
  "IssuedAt": "2026-07-29T08:00:00+00:00",
  "ExpiresIn": "2026-07-30T08:00:00+00:00", // the moment it expires
  "UserName": "agent@agency.com",
  "UserId": "...", "UserDisplayName": "...",
  "AgencyId": "...", "AgencyName": "...",
  "ContactId": "...",
  "UserType": 1, // numeric
  "ParentAgencyId": null, // null when none
  "IsAgencyOwner": false, // real boolean
  "Email": "agent@agency.com",
  "FusionSubscriptionPlan": "...",
  "FusionAgentSeats": 0,
  "FusionEnabled": true
}
```

4.3 Using the token

```
GET /api/...  
Authorization: Bearer <AccessToken>
```

THREE CHANGES THAT SILENTLY BREAK CLIENTS

- ① **Set the scheme to `Bearer` literally.** If your client builds the header by concatenating `token_type` and the token, it will now send `Authorization: JWT ...` and every request will be rejected. Hard-code `Bearer`.
- ② **`ExpiresIn` is a timestamp, not a duration.** The name carried over from OAuth but the meaning did not: it is the absolute instant the token expires. Code that computes `now + expires_in` will schedule its next refresh thousands of years out. Use `ExpiresIn` directly as the expiry, or compare it to `IssuedAt` if you need the lifetime.
- ③ **`UserType` is a number.** It was the type's name (for example `"Agent"`) and is now its numeric value. Map it rather than string-matching it.

5 Authentication response field mapping

Legacy field	New field	What to change
access_token	AccessToken	Rename only. Continue to send it as <code>Authorization: Bearer <value></code> .
token_type	TokenType	The value changes from <code>"bearer"</code> to <code>"JWT"</code> . Stop reading this field and hard-code the scheme as <code>Bearer</code> .
expires_in	ExpiresIn	ACTION Was a lifetime in seconds; is now an absolute date and time. Remove any <code>now + value</code> arithmetic.
.issued	IssuedAt	Rename, and change the parser: RFC 1123 becomes ISO 8601 with an offset.
.expires	ExpiresIn	The expiry instant now arrives under <code>ExpiresIn</code> . There is no separate <code>.expires</code> equivalent.
refresh_token	RefreshToken	Rename. The value is longer and uses a different character set, so widen any fixed-length column or validation to at least 128 characters.
userType	UserType	ACTION Type name string → number.
isAgencyOwner	IsAgencyOwner	ACTION The strings <code>"true"</code> / <code>"false"</code> → a real boolean.
parentAgencyId	ParentAgencyId	ACTION Empty string when there is no parent agency → <code>null</code> .
userName userId userDisplayName agencyId agencyName contactId email	UserName UserId UserDisplayName AgencyId AgencyName ContactId Email	Casing only. The identifiers are now real GUIDs rather than stringified ones, except <code>UserId</code> , which remains a string.
FusionSubscriptionPlan FusionAgentSeats FusionEnabled	FusionSubscriptionPlan FusionAgentSeats FusionEnabled	Previously returned only for certain sign-in flows; now always present. <code>FusionAgentSeats</code> is a number and <code>FusionEnabled</code> a boolean rather than strings.
as:client_id	—	No longer returned, because client registration is no longer part of logging in.

6 Refreshing a token

Refresh has its own route, and it needs **both** credentials: the access token you are replacing — expired is fine — and the refresh token that came with it.

Before

```
POST /api/token
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token&refresh_token=9f2c1e7a...&client_id=myApp
```

Now

```
POST /api/token/refresh
Content-Type: application/json

{
  "AccessToken": "<the token you are replacing>",
  "RefreshToken": "<the refresh token you stored>"
}
```

A successful refresh returns the same response model as login — including a **new** `RefreshToken` .

HOW TO STORE TOKENS

Every successful login and every successful refresh supersedes the previous refresh token for that user. Overwrite both values together, as one unit, and always use the newest pair. Holding on to an older refresh token will produce a `401` .

If more than one of your processes authenticates as the same Momentum AMP user, have them share one stored token pair rather than logging in independently — otherwise each login will invalidate the other's refresh token. Where processes genuinely need to be independent, give each one its own user or API key.

7 Using an API key

Previously an API key was the credential: you put it in the `Authorization` header of every request. Now you exchange it once for a token and use that token as normal.

Before

```
GET /api/...
Authorization: <api key>
```

Now - step 1, exchange the key

```
POST /api/token/exchange-api-key?apiKey=<api key>
```

Now - step 2, use the token you got back

```
GET /api/...
Authorization: Bearer <AccessToken>
```

The exchange returns the same response model as login, refresh token included, so you can keep the session alive through `/api/token/refresh` without re-sending the key.

- **Existing keys do not carry over.** Legacy keys will not validate against the Momentum AMP Public API — request new ones as part of your migration.
- **Keys can carry an expiry date** and can be revoked, so treat a `401` from the exchange as "obtain a new key", not as a transient error to retry.
- **Exchange once, then reuse.** Do not call the exchange per request; cache the token for its full lifetime and refresh it.

List endpoints

8 How list queries work — unchanged

The list endpoints are the read side of the API: collections exposed at `/api/{ListName}` and queried with standard OData options. This is the area with the fewest changes.

Aspect	Status
URL shape	UNCHANGED <code>/api/{ListName}</code> , and <code>/api/{ListName}({key})</code> for a single record
Query options	UNCHANGED <code>\$filter</code> , <code>\$select</code> , <code>\$expand</code> , <code>\$orderby</code> , <code>\$top</code> , <code>\$skip</code> , <code>\$count</code>
Property casing	UNCHANGED <code>camelCase</code> , in both responses and query expressions
Paging requirement	UNCHANGED A request must supply either a <code>\$filter</code> , or all three of <code>\$top</code> , <code>\$skip</code> and <code>\$orderby</code> . Omitting both returns <code>400</code> with a message saying so.
Write operations	List endpoints are read-only on both APIs. The new API additionally declares this in its service metadata, so tooling that reads <code>\$metadata</code> will now show the collections as not insertable, updatable or deletable. Use the REST endpoints in Part 3 to write.

```
// Works identically on both APIs
GET /api/ClaimList?$filter=changeDate gt 2026-01-01Z&$orderby=changeDate desc&$top=50&$skip=0
Authorization: Bearer <AccessToken>
```

9 List endpoints added

Momentum AMP list	What it exposes
<code>/api/PolicyChangeRequestList</code>	Policy change requests — the read side of the new endpoint area described in §12.
<code>/api/MasterCertificateList</code>	Master certificates, alongside the existing <code>CertificatesList</code> and <code>CertificateHolderList</code> .
<code>/api/IvansSessionList</code>	IVANS download session information.

10 List endpoints not available — and what to use instead

Ten legacy lists have no same-named counterpart. Four have direct replacements, and six have no equivalent on the new API.

Legacy list	Use instead	Notes
<code>/api/InsuredList</code>	<code>/api/InsuredDetailList</code>	Covers 40 of the 51 legacy fields directly and adds 21 more. The remaining fields are child collections — see §11.
<code>/api/PolicyList</code>	<code>/api/PolicyDetailList</code>	Covers 32 of the 39 legacy fields and adds 77 more. The remaining seven are child collections — see §11.

Legacy list	Use instead	Notes
/api/CustomersList	/api/InsuredDetailList	Filter on the insured type to reproduce a customers-only view. [NEEDS VERIFICATION] — confirm the field subset you rely on.
/api/SMSList	POST /api/Insured/InsuredSMS	SMS is now retrieved per insured rather than as an agency-wide queryable list. [NEEDS VERIFICATION] — confirm this covers your use case.
/api/AgencyKPI /api/WorkflowChecklistList /api/WorkflowChecklistEdit /api/WorkflowChecklistXrefList /api/ServiceRequestReplaceVehicleList /api/ExternalEmailCustomCategorySettingList	—	No equivalent on the Momentum AMP Public API. If you depend on any of these, raise it before you schedule your cut-over so it can be assessed.

11 The insured and policy list decomposition

The legacy `InsuredList` and `PolicyList` embedded child collections in every row, so a single page of results returned a deep object graph. On the new API the flat fields live on the detail list and each child collection has its own endpoint. This is the main reason list responses are smaller and faster, and it is the one change that needs real work on your side.

`InsuredList` → `InsuredDetailList` + companion endpoints

Legacy <code>InsuredList</code> field	Now retrieved from
Id	InsuredDetailList → databaseId RENAMED
NAIC	InsuredDetailList → naics RENAMED
Agents	POST /api/Insured/InsuredAgents
CSRs	POST /api/Insured/InsuredCSRs
Claims	POST /api/Insured/InsuredClaims
Policies	POST /api/Insured/InsuredPolicies
InsuredContacts	POST /api/Insured/InsuredContacts
AppliedTags	POST /api/Insured/InsuredTags
XDatesAndLinesOfBusiness	POST /api/Insured/InsuredXDatesAndLinesOfBusiness
AgencyLocation	POST /api/Insured/InsuredAgencyLocation
IsSuperVisior	Not available.
All other 40 fields	InsuredDetailList SAME NAME

`PolicyList` → `PolicyDetailList` + companion endpoints

Legacy <code>PolicyList</code> field	Now retrieved from
Agents	POST /api/Policy/PolicyAgents
CSRs	POST /api/Policy/PolicyCSR
Drivers	POST /api/Policy/PolicyDrivers
Vehicles	POST /api/Policy/PolicyVehicles
Properties	POST /api/Policy/PolicyProperties

Legacy PolicyList field	Now retrieved from
LinesOfBusiness	POST /api/Policy/PolicyLineOfBusiness
AgencyLocation	POST /api/Policy/PolicyAgencyLocation
All other 32 fields	PolicyDetailList SAME NAME (plus 77 additional fields)

RECOMMENDED PATTERN

Page the detail list first with `$filter`, `$orderby`, `$top` and `$skip`, then call only the companion endpoints whose data you actually need, passing the identifiers from that page. Most integrations discover they were paying for child collections they never read.

PART 3

REST endpoints

12 REST endpoints added

Thirty-six documented endpoints exist only on the Momentum AMP Public API, beyond the three authentication endpoints in Part 1.

Area	Endpoints
Policy change requests 13 endpoints	POST, PUT /api/PolicyChangeRequests · GET /api/PolicyChangeRequests/GetEdit · GET /api/PolicyChangeRequests/GetXmlDataInfo · POST /api/PolicyChangeRequests/ApplyChanges · POST /api/PolicyChangeRequests/Clone · POST /api/PolicyChangeRequests/FillAcord175 · POST /api/PolicyChangeRequests/BatchAssignObjects · POST /api/PolicyChangeRequests/Tags/ApplyToGeneric · POST /api/PolicyChangeRequests/Tags/BatchApplyToGeneric · POST /api/PolicyChangeRequests/Tags/BatchRemoveFromGeneric · PUT /api/PolicyChangeRequests/Tags/BatchReplaceToGeneric · POST /api/PolicyChangeRequests/Tags/BatchCleanDuplicatesForGeneric · DELETE /api/PolicyChangeRequests/Tags/DeleteFromGeneric
Carriers and MGAs 6 endpoints	POST /api/Carrier/Insert · DELETE /api/Carrier/archive · DELETE /api/Carrier/delete · POST /api/mga/Insert · DELETE /api/Mga/archive · DELETE /api/Mga/delete
Zapier operations 7 endpoints	POST /api/Zapier/InsertClaimBulk · POST /api/Zapier/InsertTaskBulk · POST /api/Zapier/InsertInsuredTagBulk · POST /api/Zapier/InsertPolicyTagBulk · POST /api/Zapier/InsertDriverViolation · POST /api/Zapier/InsertTaskCategory · POST /api/Zapier/UpdateQuoteToPolicy
Agency file management 4 endpoints	GET /api/Agency/GetAgencyFilesList · GET /api/Agency/GetAgencyFilesFolder · POST /api/Agency/RenameFileOrFolder · POST, PUT /api/Agency/UploadAgencyFilesFolderFile
Certificates 3 endpoints	POST /api/SendCertificate/PreviewCertificate · GET /api/SendCertificate/DownloadPreview/{historicalCertificateHolderId} · GET /api/MasterCertificateList/Coverages/{certificateId}
Other 3 endpoints	POST, PUT /api/GeneralLiabilities · POST /api/Tasks/UpdateAssignedTo · GET /api/sms/get-mms-file/{agencyId}/{fileId}

13 REST endpoints not available

A small number of legacy REST paths have no same-named counterpart. Four are replaced by a list query with a filter you control; the rest need a conversation before you cut over.

Replaced by a list query with `$filter`

Legacy endpoint	Use instead
GET /api/Customers/GetCustomers	GET /api/InsuredDetailList?\$filter=...
GET /api/Policy/FindPolicies	GET /api/PolicyDetailList?\$filter=...
GET /api/Driver/FindDrivers	GET /api/DriverList?\$filter=...
GET /api/Property/FindProperties	GET /api/PropertyList?\$filter=...

No equivalent — tell us if you use these

Legacy endpoint	Notes
POST /api/Receipt/TakePaymentBatch	There is no batch payment endpoint. POST /api/Receipt/InsertOrUpdateTakePayment handles one payment per call. [NEEDS VERIFICATION] — confirm whether a batch equivalent is planned.
POST /api/Zapier/InsertSMSes	No inbound SMS-creation endpoint. POST /api/Insured/InsuredSMS reads SMS for an insured.
POST /api/WorkflowChecklists/Insert PUT /api/WorkflowChecklists/Update PUT /api/WorkflowChecklists/UpdateContacts PUT /api/WorkflowChecklists/UpdateTruckingCompaniesAndPolicies GET /api/WorkflowChecklists/Xrefs	The workflow-checklist area is not present, matching the absence of its three list endpoints in \$10.
POST /api/Activities/GetViewHistory	No equivalent.

14 One verb change

Endpoint	Legacy	Now	Action
/api/Insured/UploadInsuredFile	PUT	POST or PUT	None — your existing PUT continues to work.

Every other endpoint present on both APIs accepts the same verb at the same path — including the whole customer and insured area, the whole policy area, and the full Zapier set.

Reference

15 Response codes

Code	When	How to handle it
200	The request succeeded	On authentication calls, store <code>AccessToken</code> and <code>RefreshToken</code> together and use <code>ExpiresIn</code> as the expiry instant.
400	The request is malformed, a required parameter is missing, a list query is invalid, or a list request supplied neither a <code>\$filter</code> nor all of <code>\$top</code> / <code>\$skip</code> / <code>\$orderby</code>	Body is <code>{ "message": "...", "Errors": { ... } }</code> , or for model validation <code>{ "Message": "Validation failed", "Errors": { ... } }</code> with a per-field map. Fix the request; do not retry unchanged.
401	Wrong credentials, an invalid or superseded refresh token, an invalid or expired API key, or a missing or expired access token	CHANGED The legacy API answered 403 with <code>error</code> and <code>error_description</code> . There is now no body on a failed login — stop reading those fields. Re-authenticate from the beginning rather than retrying the refresh.
404	The referenced record does not exist	Body is <code>{ "message": "...", "Errors": { ... } }</code> . Do not retry.
429	Too many requests in a short window	Back off and retry. Cache your token for its full 24-hour lifetime instead of logging in per request or per batch.
500	Unexpected server error	Body is <code>{ "message": "Internal server error", "Errors": { ... } }</code> . Retry once, then report it with the timestamp and request path.
503	The service is momentarily busy	Body is <code>{ "message": "The service is temporarily busy. Please try again.", "Errors": { ... } }</code> . Safe to retry after a short delay.

16 Migration checklist

Authentication

1. **Get your new environment details** — host names and, if you use API keys, newly issued keys.
2. **Switch the login request to JSON** and drop `grant_type`, `client_id`, `client_secret` and `scope`.
3. **Re-map the authentication response** using §5, paying particular attention to `ExpiresIn`, `UserType`, `IsAgencyOwner` and `ParentAgencyId`.
4. **Hard-code the Bearer scheme** and stop deriving it from the response.
5. **Repoint refresh** to `/api/token/refresh` and send both tokens.
6. **Store the token pair as one unit** and always overwrite it wholesale after a login or refresh.
7. **Replace API-key header usage** with the exchange call, and cache the resulting token.

Data endpoints

8. **Leave your unchanged REST and list calls alone.** They need no edits beyond the new host and token.
9. **Repoint InsuredList and PolicyList** to the detail lists, and pull child collections from the companion endpoints in §11. Rename `Id` to `databaseId` and `NAIC` to `naics`.
10. **Replace the four Find... / GetCustomers calls** with a list query and `$filter` (§13).
11. **Check the §10 and §13 "no equivalent" lists** against your integration and raise anything you depend on before scheduling the cut-over.

12. **Handle 401 where you previously handled 403**, and remove any parsing of `error` / `error_description`.
13. **Add 429 and 503 handling** with a short back-off.
14. **Verify against the staging environment** before switching production traffic: `https://staging-api.momentumamp.com/`

17 Getting help

If you hit something this guide does not cover — an endpoint whose behaviour differs from the legacy equivalent, or a response field you depended on that you cannot find — raise it with your Momentum AMP contact. Include the request path, the timestamp and the response code you received, and note whether the same call succeeds against the staging environment. That is usually enough to resolve it on the first reply.