HTTP details

Astra Automation

David Peterson
June 24, 2021

Table of Contents

HTTP details 1
API transactions and the CRUD model 1
HTTP methods 1

HTTP details

The Astra REST API uses HTTP and related parameters to act on the resources and collections. Details of the Astra HTTP implementation are presented below.

API transactions and the CRUD model

The Astra REST API implements a transactional model with well-defined operations and state transitions.

Request and response API transaction

Every REST API call is performed as an HTTP request to the Astra service. Each request generates an associated response back to the client. This request-response pair can be considered an API transaction.

Support for CRUD operational model

Each of the resource instances and collections available through the Astra REST API is accessed based on the **CRUD** model. There are four operations, each of which maps to a single HTTP method. The operations include:

- Create
- Read
- Update
- Delete

For some of the Astra resources, only a subset of these operations is supported. You should review the API reference for more information about a specific API call.

HTTP methods

The HTTP methods or verbs supported by the API are presented in the table below.

Method	CRUD	Description
GET	Read	Retrieves object properties for a resource instance or collection. This is considered a **list** operation when used with a collection.
POST	Create	Creates a new resource instance based on the input parameters. The long-term URL is returned in a `Location` response header.
PUT	Update	Updates an entire resource instance with the supplied JSON request body. Key values that are not user modifiable are preserved.
DELETE	Delete	Deletes an existing resource instance.

Request and response headers

The following table summaries the HTTP headers used with the Astra REST API.

See RFC 7232 and RFC 7233 for more information.

Header	Type	Usage notes
Accept	Request	If the value is "/" or is not provided, `application/json` is returned in Content-Type response header. If the value is set to the Astra resource Media Type, the same Media Type is returned in the Content-Type header.
Authorization	Request	Bearer token with the API key for the user.
Content-Type	Response	Returned based on the `Accept` request header.
Etag	Response	Included with a successful as defined with RFC 7232. The value is a hexadecimal representation of the MD5 value for the entire JSON resource.
If-Match	Request	A precondition request header implemented as described in section 3.1 RFC 7232 and support for **PUT** requests.
If-Modified-Since	Request	A precondition request header implemented as described in section 3.4 RFC 7232 and support for **PUT** requests.
If-Unmodified-Since	Request	A precondition request header implemented as described in section 3.4 RFC 7232 and support for **PUT** requests.
Location	Response	Contains the full URL of the newly created resource.

Query parameters

The following query parameters are available for use with resource collections. See Working with collections for more information.

Query parameter	Description
include	Contains the fields that should be returned when reading a collection.
filter	Indicates the fields that must match for a resource to be returned when reading a collection.
orderBy	Determines the sort order of resources returned when reading a collection.
limit	Limits the maximum number of resources returned when reading a collection.
skip	Sets the number of resources to pass over and skip when reading a collection.
count	Indicates if the total number of resources should be returned in the metadata object.

HTTP status codes

The HTTP status codes used by the Astra REST API are described below.

The Astra REST API also uses the **Problem Details for HTTP APIs** standard. See Diagnostics and support for more information.

Code	Meaning	Description
200	OK	Indicates success for calls that do not create a new resource instance.

Code	Meaning	Description
201	Created	An object is successfully created and the location response header includes the unique identifier for the object.
204	No content	The request was successful although no content was returned.
400	Bad request	The request input is not recognized or is inappropriate.
401	Unauthorized	The user is not authorized and must authentiate.
403	Forbidden	Access is denied due to an authorization error.
404	Not found	The resource referred to in the request does not exist.
409	Conflict	An attempt to create an object failed because the object already exists.
500	Internal error	A general internal error occurred at the server.
503	Service unavailable	The service is not ready to handle the request for some reason.