Vector POD Retrieval API Access

Table of Contents

Overview

This document introduces how to work with the Vector set of APIs to ingest images and metadata from created content.

Vector uses a metadata-driven, multi-tenant architecture to allow for rich domain modeling where new types and behavior can be added or updated at runtime.

Entity's are objects described by json schema. Everything is an entity. They may refer to each other through edges (foreign keys).

Entity Schema is the type definition, itself being an entity object. This describes the data model, validation requirements, behavioral annotations, and the user interface.

Core platform types include firms, users, views, settings, trips, documents, shares, comments, changeSets, storyboards, and so on. Platform and customer-specific types can be added or updated at runtime and backwards compatibility is enforced.

Server-side logic may be attached by the schema type or a value definition (triggers, component). The logic may be specified to run on the http request or asynchronously for post-processing, where the entity is locked while busy. Subscription rules may be used to recompute an entity when another entity is created or updated (data dependencies). Server logic can leverage relational storage for business queries and participate in the transaction for atomic writes. Typically transactions are scoped to an individual entity and subscriptions allow for asynchronous reactions, though cross-entity transactions are supported for batch operations within the http request (e.g. to allow for circular references between type definitions).

Client-side logic can be attached by ui schema, where javascript formulas are evaluated under a strict sandbox. Clients use an offline queue to synchronize due to network loss or failures and can leverage a local data store.

Storyboards are client-biased, distributed workflows that coordinate complex interactions across multiple entities and allow for users to collaborate concurrently. A plan defines the workflow and an execution is the context of what happened. Consistency is maintained through operational transformation so that users quiesce to a shared state.

Authentication

The following user has been created within the customer firm. The token is a long lived token that needs to be provided in all subsequent requests for authentication. This can be seen in the Postman collection.

Username	Password	Token
TBD	TBD	TBD

Schemas

In order to understand Vector's API calls, some understanding of Vector's metadata driven approach to entity structure is required first.

Vector uses a metadata-driven architecture, described by JsonSchema (draft 4). In this approach, most resources are operated on by a single set of CRUD routes and the resource declares the schemas that it implements. A schema defines custom properties, required fields, etc.

The Vector schemas make active use of mixins which can be thought of as a form of object inheritance. Any defined mixin within a JSON schema declares the subsequent properties that are relevant within that JSON, as a succession of additive objects.

As an example below, the Entity and Document mixin are common to all schema definitions, while the Bill of Lading mixin defines additional fields and properties for the Bill of Lading Document Type e.g.,

```
"mixins": {
 "active": [
  {
   "displayName": "Entity",
   "entityId": "11111111-0000-0000-0000-000000000000"
  },
  {
   "displayName": "Document",
   "entityId": "11111111-0000-0000-0000-000000000011"
  },
  {
   "displayName": "Bill of Lading",
   "entityId": "11111111-0000-0000-0000-000000000026"
  }
 ]
},
```

Common schemas include:

JsonSchema Id	**Entity UUID**	**Description**
entity.json	11111111-0000-0000-0000-000000000000	Abstract object model
document.jsom	11111111-0000-0000-0000-000000000011	Abstract document model
billOfLading.json	11111111-0000-0000-0000-000000000026	Bill Of Lading document type

In addition, Vector makes heavy use of UUIDs to uniquely identify every record in our system. This is also reflected when working with our APIs.

API Calls

Vector provides the following API calls which are documented below:

- POST Document - Upload
- POST Document - Share
- POST Document - Search
- GET Document - Single

Common to all API calls is the need for a bearer token:

```
--header 'Authorization: Bearer e3fa419da848a121220010819232226e1' \
```

For the POST calls only you must set the application content as JSON:

```
--header 'Content-Type: application/json' \
```

POST Document Upload - Request

This API call is designed to upload a file into the Vector platform. It is a form encoded call that requires both the file itself and the entity schema JSON to describe the file.

A POST will create the entity.
A PUT will create or update the entity. A 403 Conflict is returned if concurrently modified.

```
curl --request POST \
  --url https://api.withvector.com/1.0/entities/records \
  --header 'Content-Type: application/json' \
  --header 'content-type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW' \
  --form file:9164b4b6-c8fb-4a8a-a7f2-911252bab854=@/Users/nathancreswell/Documents/131187.pdf \
  --form 'entity={
  "mixins": {
   "active": [
    {
     "entityId": "11111111-0000-0000-0000-000000000000",
     "displayName": "Entity"
    },
    {
      "entityId": "11111111-0000-0000-0000-000000000011",
      "displayName": "Document"
    },
    {
      "entityId": "11111111-0000-0000-0000-000000000026",
      "displayName": "Bill of Lading"
    }
  ],
  "inactive": []
 },
 "uniqueId": "1a153811-a0a0-4ebf-9b67-3e6a72ce5257",
 "entity": {},
 "document": {
  "name": "nathan131test",
  "processed": false,
  "address": {
   "geolocation": {
    "longitude": -132,
    "latitude":22
   },
   "streetAddress": "",
   "locality": "",
   "region": "",
   "postalCode": "",
   "timezoneId": ""
  },
  "attachments": {
   "files": [
    {
     "name": "131187.pdf",
     "uniqueId": "9164b4b6-c8fb-4a8a-a7f2-911252bab854"
    }
   ]
  }
 },
 "billOfLading": {}
}
```

File

```
--form file:9164b4b6-c8fb-4a8a-a7f2-911252bab854=@/Users/nathancreswell/Documents/131187.pdf \
```

The file parameter in the header, if used, requires a UUID that matches the file property in the attachment JSON. This is not required if you use the URI parameter in attachments. Only if you intend to send a local file in the request. The second portion of the --from file: string is the path to the file on the user's local machine.

Unique ID

```
"uniqueId": "1e472093-e838-40d8-836f-ca576bd2a256",
```

The unique id individually identifies this document in our Vector platform. It can be used within a URL of the web browser to navigate directly to the document.

Mixins

```
"mixins": {
 "active": [
  {
   "entityId": "11111111-0000-0000-0000-000000000000",
   "displayName": "Entity"
  },
  {
   "entityId": "11111111-0000-0000-0000-000000000011",
   "displayName": "Document"
  },
  {
   "entityId": "11111111-0000-0000-0000-000000000026",
   "displayName": "Bill of Lading"
  }
```

The Document and Entity mixin will be common to every request. The Bill of Lading mixin indicates the structure of the subsequent entity section. Different document types will require a different Entity mixin in this request. Our standard document types and their Entity UUIDs are below:

DisplayName	**EntityId**
Bill of Lading	11111111-0000-0000-0000-000000000026
Freight Bill	11111111-0000-0000-0000-000000000039
Fuel Advance	11111111-0000-0000-0000-000000000040
Invoice	11111111-0000-0000-0000-000000000051
Lumper Receipt	11111111-0000-0000-0000-000000000055

Map	11111111-0000-0000-0000-000000000059
OSD	11111111-0000-0000-0000-000000000064
Other	11111111-0000-0000-0000-000000000065
Photograph	11111111-0000-0000-0000-000000000077
Pick Ticket	11111111-0000-0000-0000-000000000078
Proof of Delivery	11111111-0000-0000-0000-000000000080
Purchase Order	11111111-0000-0000-0000-000000000081
Rate Confirmation	11111111-0000-0000-0000-000000000082
Tank Wash Ticket	11111111-0000-0000-0000-000000000090
Three Prior	11111111-0000-0000-0000-000000000092
Weight Tag	11111111-0000-0000-0000-000000000101

If a customer wishes to upload a custom document type, the custom entity id of that document type needs to be passed into the API request.

Entity

```
"entity": {},
 "document": {
  "name": "nathan131test",
  "processed": false,
  "address": {
   "geolocation": {
    "longitude": -132,
    "latitude":22
   },
   "streetAddress": "",
   "locality": "",
   "region": "",
   "postalCode": "",
   "timezoneId": ""
  },
```

The entity structure is defined by the mixin referred to in the mixins section. A standard document type at a minimum can have the following fields specified:

- Name - Document name. Referenced as Load Number in our standard UI.
- Processed - true/false indicator for use in back office workflows, default to false

Address is not a required property of the document so it can be omitted. But if provided a lat/long or a street address needs to be provided.

- Geolocation - a latitude/longtiude of where the image was taken
- Street Address - physical address of where the image was taken

Attachments

Local File Example:

```
"attachments": {
    "files": [
     {
      "name": "131187.pdf",
      "uniqueId": "9164b4b6-c8fb-4a8a-a7f2-911252bab854"
     }
    ]
  },
```

URI File Example:

```
  "attachments": {
   "files": [
    {
     "uniqueId": "2bba2178-1228-48d3-99ab-d80079aa480d",
     "name": "1.jpg",
     "uri" : "https://www.gstatic.com/webp/gallery/1.jpg"
    }
   ]
  }
},
```

The attachments JSON defines the metadata around the attachment file itself. You can provide the attachment as either a multi part local file, or you can provide it via a URI parameter. You can upload multiple attachments to one document.

- Name: Filename
- UniqueId: UUID of the file itself. **Referenced in the File parameter in the header if a local file. Must be the same.**

- URI: Location of stored file. No need for File in Header if this is used.

POST Document Upload - Response

A 200 Success will include the entity as the response as per below.

A 403 Conflict is returned if it already exists or if concurrently modified.

Note that return response includes:

- The UUID of the document
- Resolved address details
- Image file locations for both preview images and the original image
- The state of processing for the document in Vector, which will most often be “busy” in this response, reflecting that operations are still being performed on the document by the Vector server
    - If “idle” then operations have ceased on the document and it can be modified

```
{
  "uniqueId": "97cebfb6-3727-4cf3-9535-ce27e547114c",
  "mixins": {
    "active": [
      {
        "entityId": "11111111-0000-0000-0000-000000000000",
        "displayName": "Entity"
      },
      {
        "entityId": "11111111-0000-0000-0000-000000000011",
        "displayName": "Document"
      },
      {
        "entityId": "11111111-0000-0000-0000-000000000026",
        "displayName": "Bill of Lading"
      }
    ],
    "inactive": []
  },
  "owner": {
    "user": {
      "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
      "displayName": "Nathan APIUser"
    },
    "firm": {
      "entityId": "c5dc8054-425a-4e5b-ba3a-24053173b0cf",
      "displayName": "Nathan's Demo Fleet"
    }
  },
  "status": {
    "state": "busy",
    "progress": {
      "percent": 0,
      "message": "Processing scheduled"
    }
  },
  "createdBy": {
    "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
    "displayName": "Nathan APIUser"
  },
  "creationDate": "2019-07-19T18:41:10.928Z",
  "modifiedBy": {
    "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
    "displayName": "Nathan APIUser"
  },
  "modifiedDate": "2019-07-19T18:41:10.928Z",
  "billOfLading": {},
  "document": {
    "name": "nathan131test",
    "processed": false,
    "address": {
      "geolocation": {
        "longitude": -132,
        "latitude": 22
      }
    },
    "attachments": {
      "files": [
        {
          "uniqueId": "9164b4b6-c8fb-4a8a-a7f2-911252bab854",
          "type": "pdf",
          "name": "131187.pdf",
          "uri":
"https://cdn.loaddocs.co/97cebfb6-3727-4cf3-9535-ce27e547114c/component/file/1563561670924/9164b4b6-c8fb-4a8a-a
7f2-911252bab854/131187.pdf"
        }
      ]
    },
    "uploadDate": "2019-07-19T18:41:10.931Z"
  },
  "entity": {}
}
```

PUT Document - Share - Request

This call will, after you have obtained the UUID of the uploaded document, allow you to share that Document with recipients. You can share with one or many recipients.

```
curl --request PUT \
 --url https://api.withvector.com/1.0/entities/records \
 --header 'Content-Type: application/json' \
 --header 'content-type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW' \
 --form 'entity={
 "mixins": {
  "active": [
   {
    "entityId": "11111111-0000-0000-0000-000000000000",
    "displayName": "Entity"
   },
   {
    "entityId": "9bfb46c8-a7ce-4e7c-bcc2-4181b4451857",
    "displayName": "Feed Event"
   },
   {
    "entityId": "11111111-0000-0000-0000-000000000022",
    "displayName": "Share"
   }
  ],
  "inactive": []
 },
 "uniqueId": "7ccd798f-f041-4ebb-8a20-ef6afc2dfd9a",
 "entity": {},
 "share": {
  "shouldSendNotification": true,
  "sharedTo": {
   "email": "illya.yanchuk@withvector.com"
  },
  "message": "Here's your Fuel Receipt",
  "entity": {
   "entityId": "68276094-4625-47cf-8280-4a5b89235b92"
  }
 }
}
```

POST Document - Share - Response

If successful, you will be returned an OK response from Vector's servers or the appropriate error code.

GET Document Single - Request

In order to retrieve a single document, the UUID needs to be provided in the URL for the GET call as per below.

```
curl --request GET \
  --url https://api.withvector.com/1.0/entities/records/1e472093-e838-40d8-836f-ca576bd2a256 \
  --header 'Authorization: Bearer e3fa419da848a12122001081923226e1' \
```

GET Document Single - Response

A 200 Success will include the entity as the response as per below.

A 404 response will be returned if the document cannot be found.

Note that return response includes:

- The UUID of the document
- Resolved address details
- Image file locations for both preview images and the original image
- The state of processing for the document in Vector, which will most often be “idle”
    - If “busy” then operations will not be able to be performed on the document

```
{
  "uniqueId": "1e472093-e838-40d8-836f-ca576bd2a256",
  "mixins": {
    "active": [
      {
        "entityId": "11111111-0000-0000-0000-000000000000",
        "displayName": "Entity"
      },
      {
        "entityId": "11111111-0000-0000-0000-000000000011",
        "displayName": "Document"
      },
      {
        "entityId": "11111111-0000-0000-0000-000000000026",
        "displayName": "Bill of Lading"
      }
    ],
    "inactive": []
  },
  "owner": {
    "user": {
      "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
      "displayName": "Nathan APIUser"
    },
    "firm": {
      "entityId": "c5dc8054-425a-4e5b-ba3a-24053173b0cf",
      "displayName": "Nathan's Demo Fleet"
    }
  },
  "status": {
    "state": "idle"
  },
  "createdBy": {
    "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
    "displayName": "Nathan APIUser"
  },
  "creationDate": "2019-07-08T22:34:52.083Z",
  "modifiedBy": {
    "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
    "displayName": "Nathan APIUser"
  },
  "modifiedDate": "2019-07-08T22:34:52.083Z",
  "billOfLading": {},
  "document": {
    "name": "131187",
    "address": {
      "region": "CA",
      "locality": "San Francisco",
      "postalCode": "94107",
      "timezoneId": "America/Los_Angeles",
      "countryName": "United States",
      "geolocation": {
        "latitude": 37.781085399999995,
        "longitude": -122.39709640000001
      },
      "streetAddress": "131 Stillman Street"
    },
    "processed": false,
    "uploadDate": "2019-07-08T22:34:52.087Z",
    "attachments": {
      "files": [
        {
          "uri":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/file/1562625292085/9164b4b6-c8fb-4a8a-
a7f2-911252bab854/131187.pdf",
          "name": "131187.pdf",
          "type": "pdf",
          "pages": 2,
          "preview": [
            {
              "uri":
"https://images.withvector.com/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/9164b4b6-c8f
b-4a8a-a7f2-911252bab854/page_1.jpg",
                "name": "page_1",
                "text":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/image/1562625300108/88a66600-5c6c-4a
42-ac24-83b94ccc258a/page1.txt",
                "type": "image",
                "width": 1699,
                "height": 2200,
                "source": [
                  {
                    "uri":
"https://images.withvector.com/fit-in/1650x1650/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/15626253001
08/9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                    "width": 1650,
                    "height": 1650
                  },
                  {
                    "uri":
"https://images.withvector.com/fit-in/640x480/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/
9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                    "width": 640,
                    "height": 480
                  },
                  {
                    "uri":
"https://images.withvector.com/fit-in/64x64/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/91
64b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                    "width": 64,
                    "height": 64
                  }
                ],
                "uniqueId": "88a66600-5c6c-4a42-ac24-83b94ccc258a"
              },
              {
                "uri":
"https://images.withvector.com/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/9164b4b6-c8f
b-4a8a-a7f2-911252bab854/page_2.jpg",
                "name": "page_2",
                "text":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/image/1562625300108/a0af1572-f40f-45d
9-ae4d-605b6dc53c2d/page2.txt",
                "type": "image",
                "width": 1699,
                "height": 2200,
                "source": [
                  {
                    "uri":
"https://images.withvector.com/fit-in/1650x1650/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/15626253001
08/9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                    "width": 1650,
                    "height": 1650
                    },
                    {
                      "uri":
"https://images.withvector.com/fit-in/640x480/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/
9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                      "width": 640,
                      "height": 480
                    },
                    {
                      "uri":
"https://images.withvector.com/fit-in/64x64/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/91
64b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                      "width": 64,
                      "height": 64
                    }
                  ],
                  "uniqueId": "a0af1572-f40f-45d9-ae4d-605b6dc53c2d"
                }
              ],
              "uniqueId": "9164b4b6-c8fb-4a8a-a7f2-911252bab854"
            }
          ],
          "aggregate": {
            "uri":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/file/1562625292085/9164b4b6-c8fb-4a8a-
a7f2-911252bab854/131187.pdf",
            "name": "aggregate.pdf",
            "type": "pdf",
            "pages": 2,
            "uniqueId": "c6b37cc8-d1f6-4863-a0af-c1779d78fa3f"
          }
        }
      },
      "entity": {}
}
```

POST Document Search - Request

This API call will take a set of parameters, perform a search, and return with the associated document metadata in a sorted, paginated order. In the example below, documents created within a 7 day period are returned in descending order with a page size of 80.

```
{
  "filters": [
    {
      "type": "containsEdge",
      "path": "mixins.active",
      "values": [
       { "entityId": "11111111-0000-0000-0000-000000000011" }
      ]
    },
    {
      "type": "range",
      "entityType": "/1.0/entities/metadata/entity.json",
      "label": "Posted Date",
      "path": "creationDate",
      "gte": "2019-06-30T07:00:00.000Z",
      "lte": "2019-07-07T06:59:59.999Z"
    }
  ],
  "orders": [
    {
      "path": "creationDate",
      "type": "descending",
      "label": "Posted Date"
    }
  ],
  "metadata": {
    "size": 80,
    "offset": 0,
    "maxSizePerGroup": 10,
    "shouldIncludeLeafEntities": true
  }
}
```

Filters

Type

The following options are available for type:

- matchedge: filter on an exact value of an edge, i.e., constrain a filter to just a single entity type such as documents

Single Value

```
{
  "type" : "matchEdge",
  "path" : "owner.user",
  "value" : {
   "entityId" : "11111111-2222-3333-4444-555555555555",
   "displayName" : "Some User Name"
  }
}
```

Multiple Values

```
{
  "type" : "match",
  "path" : "owner.user",
  "values" : [
    {
      "entityId" : "11111111-2222-3333-4444-555555555555",
      "displayName" : "Some User Name"
    },
    {
      "entityId" : "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
      "displayName" : "Other User Name"
    }
  ]
}
```

- containsedge: filter on an exact value on an array of edges, i.e., constrain a filter to just documents or other entities

Single Value

```
{
  "type" : "containsEdge",
  "path" : "mixins.active",
  "value" : {
    "entityId" : "11111111-0000-0000-0000-000000000011",
    "displayName" : "Document"
  }
}
```

Multiple Values

```
{
  "type" : "containsEdge",
  "path" : "mixins.active",
  "values" : [
    {
      "entityId" : "11111111-0000-0000-0000-000000000011",
      "displayName" : "Document"
  },
  {
   "entityId" : "11111111-0000-0000-0000-000000000209",
   "displayName" : "Broker Order"
  }
 ]
}
```

- match : filter on exact value of a field

Single Value

```
{
 "type" : "match",
 "path" : "document.name",
 "value" : "Some Document Name"
}
```

Multiple Values

```
{
 "type" : "match",
 "path" : "document.name",
 "values" : [
  "Some Document Name",
  "Other Document Name"
 ]
}
```

- match entity: filter on exact UUID value of an entity

Single Value

```
{
 "type" : "matchEntity",
 "value" : {
  "uniqueId" : "11111111-2222-3333-4444-555555555555",
   ...
   ...
   ...
 }
}
```

Multiple Values

```
{
 "type" : "matchEntity",
 "values" : [
  {
   "uniqueId" : "11111111-2222-3333-4444-555555555555",
    ...
    ...
    ...
  },
  {
   "uniqueId" : "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    ...
    ...
    ...
  }
 ]
}
```

- contains: filter on a value within an array field

Single Value

```
{
  "type" : "contains",
  "path" : "dailyInspectionReport.equipmentOnSite",
  "value" : "Equipment One"
}
```

Multiple Values

```
{
  "type" : "contains",
  "path" : "dailyInspectionReport.equipmentOnSite",
  "values" : [
    "Equipment One",
    "Equipment Two"
  ]
}
```

- prefix: filter on the prefix of a string field

Single Value

```
{
  "type" : "prefix",
  "path" : "business.legalName",
  "value" : "Proctor"
}
```

Multiple Values

```
{
  "type" : "prefix",
  "path" : "business.legalName",
  "values" : [
    "General",
    "Produce"
  ]
}
```

- range: filter on numeric or date field, parameters available are:
    - gt : greater than
    - gte : greater than or equal
    - lt : less than
    - lte : less than or equal

```
{
  "type" : "range",
  "path" : "creationDate",
  "gte" : "2016-06-01T00:00:00.000+0000",
  "lte" : "2016-11-01T00:00:00.000+0000"
}
```

Orders

Orders control the sort order of the returned response.

- type
  - ascending
  - descending
- path
  - path to field or edge
  - special syntax "_count" for sorting group by count of entities

Metadata

The final metadata parameters will return the count of size and offsets, for example.

- Size: count of items to return for top level group
- Offset: offset of items to return from for top level group
- maxSizePerGroup: count of items to return for top level groups
- shouldIncludeLeafEntities: indication to add entities as children to leaf groups (short hand convenience to avoid adding group:Entity)

POST Document Search - Response

A 200 Success will include the entity as the response as per below. Even if no results are found a 200 response with 0 records will be returned
The response for a document will return an array of matching documents with associated metadata, including image locations as per the single document GET call.

Note that there is a count of documents returned at the suffix of each response.

```
{
  "children": [
    {
      "data": {
        "uniqueId": "1e472093-e838-40d8-836f-ca576bd2a256",
        "owner": {
          "user": {
            "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
            "displayName": "Nathan APIUser"
        },
        "firm": {
          "entityId": "c5dc8054-425a-4e5b-ba3a-24053173b0cf",
          "displayName": "Nathan's Demo Fleet"
        }
      },
      "mixins": {
        "active": [
          {
            "entityId": "11111111-0000-0000-0000-000000000000",
            "displayName": "Entity"
          },
          {
            "entityId": "11111111-0000-0000-0000-000000000011",
            "displayName": "Document"
          },
          {
            "entityId": "11111111-0000-0000-0000-000000000026",
            "displayName": "Bill of Lading"
          }
        ],
        "inactive": []
      },
      "createdBy": {
        "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
        "displayName": "Nathan APIUser"
      },
      "billOfLading": {},
      "document": {
        "processed": false,
        "address": {
          "streetAddress": "131 Stillman Street",
          "locality": "San Francisco",
          "region": "CA",
          "postalCode": "94107",
          "countryName": "United States",
          "timezoneId": "America/Los_Angeles",
          "geolocation": {
            "latitude": 37.781085399999995,
            "longitude": -122.39709640000001
          }
        },
        "attachments": {
          "files": [
            {
              "uniqueId": "9164b4b6-c8fb-4a8a-a7f2-911252bab854",
              "type": "pdf",
              "name": "131187.pdf",
              "uri":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/file/1562625292085/9164b4b6-c8fb-4a8a-
```

```
a7f2-911252bab854/131187.pdf",
                    "preview": [
                      {
                        "width": 1699,
                        "height": 2200,
                        "source": [
                          {
                            "uri":
"https://images.withvector.com/fit-in/1650x1650/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/15626253001
08/9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                            "width": 1650,
                            "height": 1650
                          },
                          {
                            "uri":
"https://images.withvector.com/fit-in/640x480/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/
9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                            "width": 640,
                            "height": 480
                          },
                          {
                            "uri":
"https://images.withvector.com/fit-in/64x64/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/91
64b4b6-c8fb-4a8a-a7f2-911252bab854/page_1.jpg",
                            "width": 64,
                            "height": 64
                          }
                        ],
                        "text":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/image/1562625300108/88a66600-5c6c-4a
42-ac24-83b94ccc258a/page1.txt",
                        "uniqueId": "88a66600-5c6c-4a42-ac24-83b94ccc258a",
                        "type": "image",
                        "name": "page_1",
                        "uri":
"https://images.withvector.com/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/9164b4b6-c8f
b-4a8a-a7f2-911252bab854/page_1.jpg"
                      },
                      {
                        "width": 1699,
                        "height": 2200,
                        "source": [
                          {
                            "uri":
"https://images.withvector.com/fit-in/1650x1650/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/15626253001
08/9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                            "width": 1650,
                            "height": 1650
                          },
                          {
```

```
                        "uri":
"https://images.withvector.com/fit-in/640x480/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/
9164b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                        "width": 640,
                        "height": 480
                      },
                      {
                        "uri":
"https://images.withvector.com/fit-in/64x64/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/91
64b4b6-c8fb-4a8a-a7f2-911252bab854/page_2.jpg",
                        "width": 64,
                        "height": 64
                      }
                    ],
                    "text":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/image/1562625300108/a0af1572-f40f-45d
9-ae4d-605b6dc53c2d/page2.txt",
                    "uniqueId": "a0af1572-f40f-45d9-ae4d-605b6dc53c2d",
                    "type": "image",
                    "name": "page_2",
                    "uri":
"https://images.withvector.com/1e472093-e838-40d8-836f-ca576bd2a256/component/pdf/1562625300108/9164b4b6-c8f
b-4a8a-a7f2-911252bab854/page_2.jpg"
                  }
                ],
                "pages": 2
              }
            ],
            "aggregate": {
              "pages": 2,
              "uniqueId": "c6b37cc8-d1f6-4863-a0af-c1779d78fa3f",
              "type": "pdf",
              "name": "aggregate.pdf",
              "uri":
"https://cdn.loaddocs.co/1e472093-e838-40d8-836f-ca576bd2a256/component/file/1562625292085/9164b4b6-c8fb-4a8a-
a7f2-911252bab854/131187.pdf"
            }
          },
          "name": "131187",
          "uploadDate": "2019-07-08T22:34:52.087Z"
        },
        "modifiedDate": "2019-07-08T22:34:52.083Z",
        "modifiedBy": {
          "entityId": "b869dab6-36ba-42e1-b451-4fa1763ec8d9",
          "displayName": "Nathan APIUser"
        },
        "precomputation": {
          "entityType": {
            "entityId-11111111-0000-0000-0000-000000000011": {
              "entityId": "11111111-0000-0000-0000-000000000026",
              "displayName": "Bill of Lading"
            }
          },
          "activity": {
            "numberOfComments": 0,
            "numberOfShares": 0
          },
          "displayName": "131187"
        },
        "creationDate": "2019-07-08T22:34:52.083Z",
        "entity": {},
        "status": {
          "state": "idle"
        }
      },
      "children": [],
      "metadata": {
        "totalEntityMatchCount": 1
      }
    }
  ],
  "metadata": {
    "totalEntityMatchCount": 1,
    "totalChildrenCount": 1
  }
}
```

Timeouts

Vector’s load balancer will automatically timeout requests after 60 seconds as a default.