



Getting Started with Dovetail APIs

API Documentation

API docs are available online at <https://YourTenant.dovetailnow.com/api/doc>


Example: <https://sandbox.dovetailnow.com/api/doc>

API User

From within the Dovetail Agent application, create a new user.

A common pattern is to have the username be something easy to identify as an API user, e.g. *api*, *dovetail-api*, etc.

By creating a user explicitly for API use, this allows you to have some control and insights into activities that are done by that user. Separation of API users from standard application users allows for greater security and access control to the Dovetail application.



Create ▾Query ▾Apps ▾

Create User

First Name *	<input type="text" value="Dovetail"/>
Last Name *	<input type="text" value="API"/>
Email *	<input type="text" value="api@example.org"/>
Username *	<input type="text" value="doetail-api"/>
Site *	<input type="text" value="1001 - Dovetail Software (Corporate)"/> ✕ ▾
Job Title	<input type="text"/>
Department	<input type="text"/> ▾
Workgroup	<input type="text"/> ▾
Work Calendar	<input type="text"/> ▾
Status	<input type="text" value="Active"/> ▾

Password

Once the user is created, be sure to set the user's password, as API users need to login using a loginName and password, not via SSO.

User Status

The user must have a status of *Active*. If the status is changed to *Inactive*, the user will not be able to login and APIs will error.

User Roles

The user must belong to a role, and that role needs to be granted access to *Authenticate with API*.

Role

API User

Apply to all new users? ☐ False

Grants access to use the Dovetail API

Need Help?

A role is an entity that is granted permissions and can be assigned to a user. The role page allows you to add and edit details about a role as you manage the permissions. You can grant users permissions either as a category or in part via individual overrides.

Permissions

Legend

- Access to all permissions in the category
- Access dependent on category or exceptions
- Access to no permissions in the category

Users

Permissions

Show All

Show Assigned

Search

Apply Changes

Security and Authorization

with some exceptions

API calls can impersonate other users	Granted	Authenticate with API	Granted
Authenticate with login screen		Authenticate with SSO	

Additionally, you may wish to grant permission for *API calls can impersonate other users*. Some of the newer Dovetail APIs (create case, close case, log note, and add attachment) allow for passing in arbitrary usernames as the user performing the action. For example, the user “John” can call the log-note API, and pass in a username of "Mary", thus making it appear like the note was logged by "Mary". The “API calls can impersonate other users” permission determines whether or not a user is allowed to do this.

The API user should be added to this role.

Create ▾ Query ▾ Apps ▾ Admin ▾

Search

0

Super User ▾

User

Dovetail API

Status

Active

System Admin

API User

Username

dovetail-api

Site

Dovetail MEX

Job Title

Department

None

Workgroup

Support

Work Calendar

Email

dovetail-api@local...

Last Login

2 years ago

Settings

Queues 2

Roles 1

Role Name

API User

Tools

Let's look at a couple of common tools used for getting started and testing APIs.

curl

Curl is a command line tool for transferring data with URLs, and is a quick way to test APIs. curl is available for all major platforms, including Windows, MacOS, and UNIX. It is typically included as part of the operating system.

Postman

Postman is a tool that also allows you to call APIs. It has a user interface, which often makes it easier to understand how to make API calls, and to understand the results.

Both curl and Postman are useful to do some quick testing of APIs. But, once you understand how an API works, you'll likely want to make those API calls from a programming language, or some other tool or service.

Calling Dovetail APIs with curl

Get the details of a case

From a command line terminal on your machine, use curl to call the Dovetail API in order to retrieve the details of a case.

```
curl -u "dovetail_api:mypassword"  
https://mytenant.dovetailnow.com/api/v1/Cases/1234
```

(be sure to specify your own username, password, tenant, and case ID)

That should succeed and return a bunch of data to the screen. This data is all of the details of the case, in a JSON format. ([What is JSON?](#))

What if that fails?

If that fails, and returns nothing, then likely there's a problem with the username, password, or the status of that user.

Run that same curl command again, but this time, include the *-i* option, which tells curl to show us the response headers.

```
curl -i -u "dovetail_api:mypassword"  
https://mytenant.dovetailnow.com/api/v1/Cases/1234
```

If the first line says: HTTP/1.1 403 UNAUTHORIZED

Then the problem is likely one of:

- Invalid username
- Invalid password
- User status is not Active

If the first line says: HTTP/1.1 404 Not Found

Then the problem is likely that the case ID supplied was not found in the system

All that case data is hard to read!

If the GET case API succeeds, it will output a bunch of data to the screen. This data is all of the details of the case, in a JSON format. It can be overwhelming and difficult to read.

Rather than have all of that case data in your terminal window, you can redirect the output to a file. For example:

```
curl -u "dovetail_api:mypassword"
https://mytenant.dovetailnow.com/api/v1/Cases/1234 > case1234.txt
```

That will create a text file named *case1234.txt*. Open that file with your favorite text editor, and you will see all of the details of that case.

The JSON data that is in that file is not in a pretty format. Typically, JSON is read by a computer program, not a human. But, if you want to see it in a pretty, more readable format, then simply open that text file, copy everything to your clipboard, and paste it into a *Pretty JSON tool* such as <http://jsonprettyprint.net/>

Create a case

Now that we've retrieved the details of a case, let's create a case.

Rather than doing a GET to retrieve the details, we'll POST data to the API. (GET and POST are two of the verbs/methods that are built-in to HTTP. [Learn More here](#))

```
curl -i -u "dovetail_api:mypassword"
https://mytenant.dovetailnow.com/api/v2/Cases --request POST --header
"Content-Type: application/json" --header "Accept: application/json"
--data-raw '{"title': 'Hello from the API', 'notes': 'These are some
notes', 'employee': '12345'}"
```

There's a bunch of new stuff there, so let's break that down:

curl	The curl executable
-i	Display the headers of the response. Useful when troubleshooting
-u "dovetail_api:mypassword"	Username and Password
https://mytenant.dovetailnow.com/api/v2/Cases	The URL for the create case API

<code>--request POST</code>	Defines this as a POST, as opposed to a GET request. Learn More here
<code>--header "Content-Type: application/json"</code>	This tells the API that we're sending data in a JSON format
<code>--header "Accept: application/json"</code>	This tells the API that we expect to receive the response in a JSON format
<code>--data-raw</code> <pre>{ 'title': 'Hello from the API', 'notes': 'These are some notes', 'employee': '12345' }</pre>	This is the actual data that we're sending. Notice that it's in a JSON format. We're sending the case title, the initial notes, and the ID of the employee that the case is to be created for.

Other available options?

If we [look at the API documentation](#), we can see that we have a number of additional optional options that can be set when creating the case:

POST /v2/cases Create case ▼	
Create a case on behalf of an employee	
Request	
Json	Type/Comments
{	CreateCaseRequest
"title": "",	string Required The title of the case
"notes": "",	string Required The initial notes logged to this case
"employee": "",	string Required The Employee ID or email address of the employee associated with this case
"originator": "",	string Optional
"portalCaseType": "",	string Optional List: PortalCaseType The portal case type (the key value, not the localized display text)
"site": "",	string Optional The Site ID of the site the case should be logged against
"concerning": "",	string Optional The Employee ID or email address of the concerning employee
"severity": "",	string Optional List: CaseIssueSeverity The severity level of the case (the key value, not the localized display text)
"priority": ""	string Optional List: CasePriority The priority of the case (the key value, not the localized display text)
}	

Response Codes

Every HTTP request will return a status code, indicating success or failure, and potentially some additional information. Here are some common responses from the Dovetail APIs:

Response	Comments
200 OK or 201 Successfully Created	All is well. Depending on the specific API, the success response code could be either 200 or 201. A 201 is commonly used to indicate that something was successfully <i>created</i> (such as a new case).
500 Internal Server Error	Typically, this means something went wrong on the API side. You will also get this error if the employee ID passed in does not match to an employee in the Dovetail system.
400 Bad List Value	This means one of the list values (Severity, Priority, or PortalCaseType) that you passed in is invalid. Check the <i>X-Dovetail-BadListValue</i> header to see which list it is.
400 Value is Required	This means one of the required parameters (title, notes, or employee) was not supplied. Check the <i>X-Dovetail-MissingParameter</i> header to see which parameter it is
400 Invalid Parameter	Invalid parameter. Check the <i>X-Dovetail-InvalidParameter</i> header to see which parameter it is
403 UNAUTHORIZED	The problem is likely one of: <ul style="list-style-type: none">• Invalid username• Invalid password• User status is not Active
404 Not Found	This can happen if: <ul style="list-style-type: none">• The URL supplied for the API is incorrect• The case ID provided is not correct when retrieving a case by ID.

Rinse, Lather, Repeat

Follow the same patterns as above for using the other [available Dovetail APIs](#)

Calling Dovetail APIs with Postman

Rather than using curl from the command line, we can use Postman, a free app with a helpful user interface that is often easier to use than a command line.

Get the details of a case

- Download and Install the [Postman app](#)
- Within the app, from the *New* menu in the upper-left, choose *New - Request*
- Give it a name, such as *Get a dovetail case*
- Optionally put it into a collection/folder. Here, it's being saved to a collection called *Dovetail API testing*
- Click *Save*

SAVE REQUEST

Requests in Postman are saved in collections (a group of requests).
[Learn more about creating collections](#)

Request name

Get a dovetail case

Request description (Optional)

Adding a description makes your docs better

Descriptions support Markdown

Select a collection or folder to save to:

Q Search for a collection or folder

◀ Dovetail API Testing

+ Create Folder

GET Get case 1053

Cancel

Save to Dovetail API Testing

This will create a new tab where we can continue to define the request.
Enter the URL to retrieve a case, just as we did earlier. e.g.
`https://mytenant.dovetailnow.com/api/v1/Cases/1234`

Note: the last part of that URL (1234) is the ID number of the case that we wish to retrieve. That should be a valid case in your system.

GET Get a dovetail case

+

...

▶ Get a dovetail case

GET

https://dev.dovetailnow.com/api/v1/Cases/1053

Params

Authorization ●

Headers

Body

Pre-request Script

Tests

KEY	VALUE
Key	Value

Response

1. Click on the Authorization tab
2. Select Basic Auth from the Type dropdown
3. Enter the username
4. Enter the password

GET Get a dovetail case

+

...

▶ Get a dovetail case

GET

https://dev.dovetailnow.com/api/v1/Cases/1053

Params

Authorization ●

Headers

Body

Pre-request Script

Tests

TYPE

Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Preview Request

Username

dovetail_api

Password

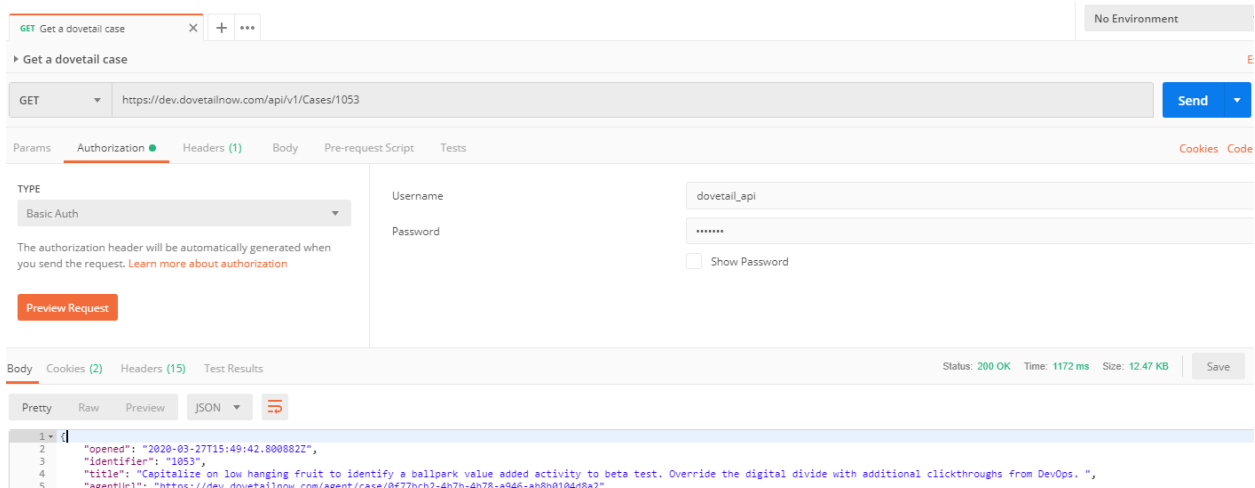
.....

☐ Show Password

Response

1. Click the Save button, so that our work is saved.

2. Click on the Send button. This will send the request, and the lower portion of the screen will show the response. If all is well, it will show the details of the case, in a pretty JSON format.



Take Note of:

- The Status is displayed (in this case **200 OK**)
- In addition to the Body of the response, there is also a Headers tab, so we can investigate the response headers, which is helpful when troubleshooting a problem.

Create a case

Now that we've retrieved the details of a case, let's create a case.

Rather than doing a GET to retrieve the details, we'll POST data to the API. (GET and POST are two of the verbs/methods that are built-in to HTTP. [Learn More here](#))

Basics

- Create a new request, as before, but this time we'll name it *Create a case*
- Change the method dropdown to POST
- Enter the URL for the API to create a case, e.g.
https://mytenant.dovetailnow.com/api/v2/Cases

Authorization

- Click on the Authorization tab
- Select Basic Auth from the Type dropdown
- Enter the username
- Enter the password

► Create a case

POST ▼ https://dev.dovetailnow.com/api/v2/Cases

Params Authorization ● Headers (1) Body Pre-request Script Tests

TYPE

Basic Auth ▼

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username

Password

☐ Show Password

Headers

- Click on the Headers tab
- Add a new key of “Content-Type”
- Add a value of application/x-www-form-urlencoded

► Create a case

POST ▼ https://dev.dovetailnow.com/api/v2/Cases

Params Authorization ● Headers (1) Body Pre-request Script Tests

KEY	VALUE
<input checked="" type="checkbox"/> Content-Type	application/x-www-form-urlencoded
Key	Value

Response

Body

- Click on the Body tab
- Select the x-www-form-urlencoded radio button
- Add in the key/value pairs for the data.
 - Title
 - Notes
 - Employee

► Create a case

POST ▼ https://dev.dovetailnow.com/api/v2/Cases

Params Authorization ● Headers (1) **Body ●** Pre-request Script Tests

● none ● form-data ● x-www-form-urlencoded ● raw ● binary

KEY	VALUE
<input checked="" type="checkbox"/> Title	Creating a case from Postman using the Dovetail API
<input checked="" type="checkbox"/> Notes	these are notes
<input checked="" type="checkbox"/> Employee	12221
Key	Value

Response

Save & Send

- Click the **Save** button, so that our work is saved.
- Click on the **Send** button. This will send the request, and the lower portion of the screen will show the response. If all is well, it will show the details of the newly created case, in a pretty JSON format.

POST Create a case × + ... No Environment 🔍

► Create a case Examples (0) ▼

POST ▼ https://dev.dovetailnow.com/api/v2/Cases Send ▼ Save ▼

Params Authorization ● Headers (2) **Body ●** Pre-request Script Tests Cookies Code Comments (0)

● none ● form-data ● x-www-form-urlencoded ● raw ● binary

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Title	Creating a case from Postman using the Dovetail API	
<input checked="" type="checkbox"/> Notes	these are notes	
<input checked="" type="checkbox"/> Employee	12221	
Key	Value	Description

Body Cookies (2) Headers (14) Test Results Status: 201 Successfully Created Time: 952 ms Size: 3.33 KB Save Download

Pretty Raw Preview JSON 🔍

```

1 {
2   "opened": "2020-05-29T18:48:34.588971Z",
3   "identifier": "1881",
4   "title": "Creating a case from Postman using the Dovetail API",
5   "agentUrl": "https://dev.dovetailnow.com/agent/case/f0d7d93f-c64f-44d9-9bdd-abca0135f8ee",
6   "selfServiceUrl": "https://dev.portal.demo.dovetailnow.com/case/f0d7d93f-c64f-44d9-9bdd-abca0135f8ee",
7   "id": "f0d7d93f-c64f-44d9-9bdd-abca0135f8ee",
8   "condition": {
9     "listName": "CaseCondition",
10    "key": "Open",
11    "displayText": "Open"
12  },
13  "issueSeverity": {
14    "listName": "CaseIssueSeverity",
15    "key": "Medium",
16    "displayText": "Medium"
17  },
18  "employee": {
19    "id": "Bf9411cf-2eb3-4d78-a99c-ab68817c88df",
20    "avatarUrl": "https://s3-us-west-2.amazonaws.com/avatars.demo.us-west-2.dovetailnow.com/dev/00000000-0000-0000-0000-000000000000",
21    "employeeId": "12221",
22    "title": null,
23    "description": null
24  }
25 }
```

Take Note of

- The Status is displayed (in this case **201 Successfully Created**)
- In addition to the Body of the response, there is also a Headers tab, so we can investigate the response headers, which is helpful when troubleshooting a problem.

Rinse, Lather, Repeat

Follow the same patterns as above for using the other [available Dovetail APIs](#)

Additional Notes

For **multi-line text fields**, such as *notes*, use “\n” to indicate a newline.

Example: "This is line 1 of notes.\nThis is line 2 of notes",

Common Problems:

Status: 400 Value is Required

This can happen when the form data parameter does not have the proper casing. Notice that this request failed, because the form data uses a key of *title* instead of *Title*

The screenshot shows a Postman interface for a POST request to `https://dev.dovetailnow.com/api/v2/Cases`. The request body is form-encoded with the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> title	Creating a case from Postman using the Dovetail API	
<input checked="" type="checkbox"/> Notes	these are notes	
<input checked="" type="checkbox"/> Employee	12221	
Key	Value	Description

The response status is **400 Value is Required**. The response headers include:

- Content-Length → 0
- Connection → keep-alive
- Cache-Control → private
- Date → Fri, 29 May 2020 18:50:46 GMT
- Server →
- Strict-Transport-Security → max-age=31536000; includeSubDomains; preload
- X-Dovetail-MissingParameter → Title**
- X-Frame-Options → SAMEORIGIN

Status: 400 Bad List Value

This means one of the list values (Severity, Priority, or PortalCaseType) that you passed in is invalid. Check the *X-Dovetail-BadListValue* header to see which list it is.

The screenshot shows a Postman interface for a POST request to `https://dev.dovetailnow.com/api/v2/Cases`. The request body is in the 'x-www-form-urlencoded' format with the following data:

KEY	VALUE	DESCRIPTION
Title	Creating a case from Postman using the Dovetail API	
Notes	these are notes	
Employee	12221	
Severity	foo	

The response status is **400 Bad List Value**. The headers section shows the following response headers:

- Content-Length → 0
- Connection → keep-alive
- Cache-Control → private
- Date → Fri, 29 May 2020 18:53:59 GMT
- Server →
- Strict-Transport-Security → max-age=31536000; includeSubDomains; preload
- X-Dovetail-BadListValue → Severity**
- X-Frame-Options → SAMEORIGIN

Status: 403 Forbidden

Normally, this is one of:

- Invalid username
- Invalid password
- User status is not Active

In addition, this can happen when using GET instead of POST when trying to create a case, as shown here:

GET Create a case

GET Get case 1053

+ ...

No Environment

Create a case

Examples (0)

GET

https://dev.dovetailnow.com/api/v2/Cases

Send

Save

ParamsAuthorizationHeaders (2)BodyPre-request ScriptTests

CookiesCodeComments (0)

noneform-datax-www-form-urlencodedrawbinary

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	Title	Creating a case from Postman using the Dovetail API			
<input checked="" type="checkbox"/>	Notes	these are notes			
<input checked="" type="checkbox"/>	Employee	12221			
	Key	Value	Description		

BodyCookies (2)Headers (9)Test Results

Status: 403 ForbiddenTime: 283 msSize: 1.24 KB

SaveDownload

PrettyRawPreviewHTML

1

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

2

<HTML>

3

<HEAD>

4

<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">

5

<TITLE>ERROR: The request could not be satisfied</TITLE>

6

</HEAD>

7

<BODY>

8

<H1>403 ERROR</H1>

9

<H2>The request could not be satisfied.</H2>

10

<HR noshade size="1px">

11

Bad request.

12

We can't connect to the server for this app or website at this time. There might be too much traffic or a configuration error. Try again later, or contact the app or website owner.

13

14

<BR clear="all">

15

If you provide content to customers through CloudFront, you can find steps to troubleshoot and help prevent this error by reviewing the CloudFront documentation.

16

17

<BR clear="all">

18

<HR noshade size="1px">

19

<PRE>

20

Generated by cloudfront (CloudFront)

21

Request ID: YpY2bk_PaCEWjwMYt4l7Yjb9wcFnt07z332e-Zba8vpk598Z8d6VQ==

22

</PRE>

23

<ADDRESS></ADDRESS>

24

</BODY>

25

</HTML>