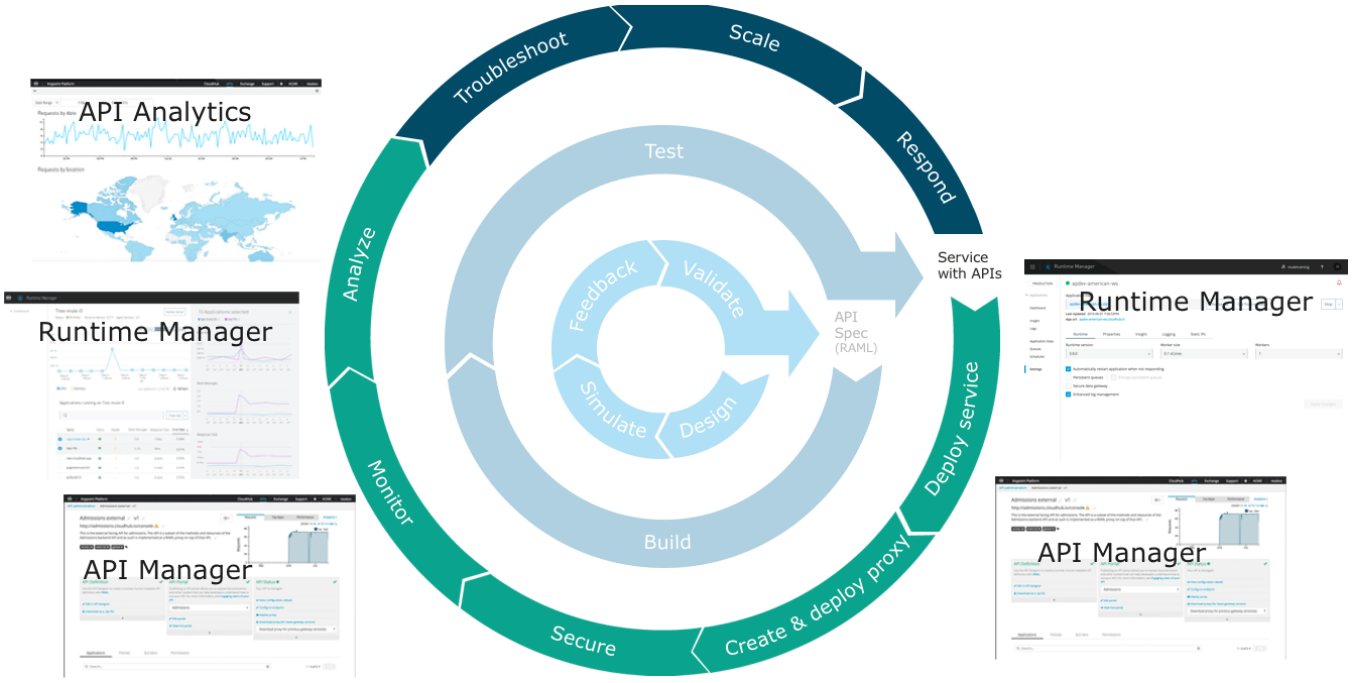
Module 5: Deploying and Managing APIs



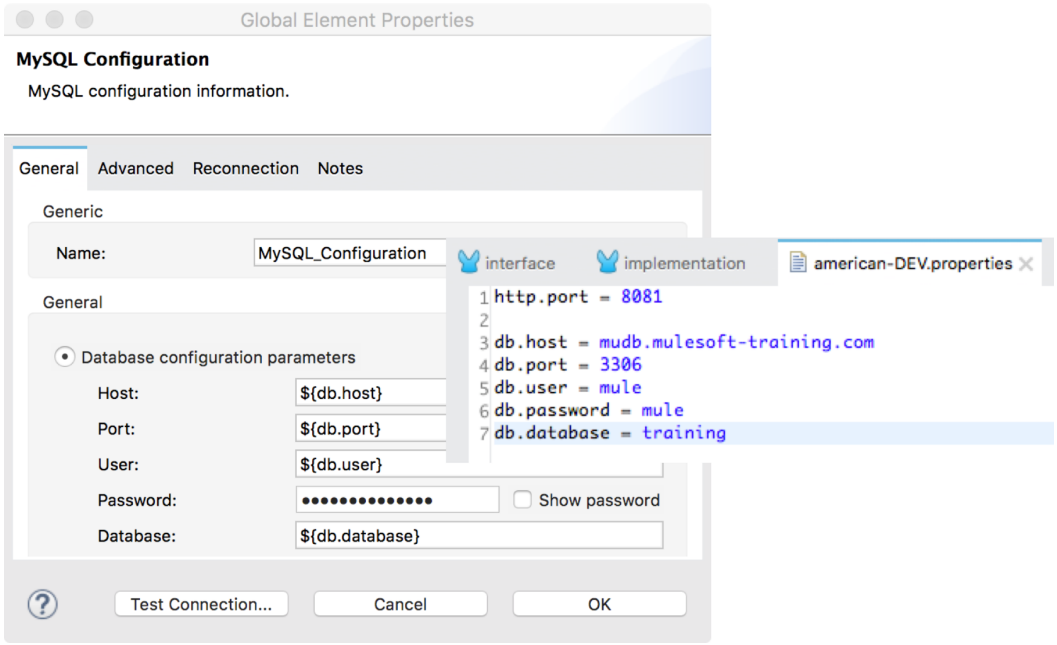
At the end of this module, you should be able to:

- Describe the options for deploying Mule applications.
- Use properties in Mule applications so they can be easily moved between environments.
- Deploy Mule applications to CloudHub.
- Use API Manager to create and deploy API proxies.
- Use API Manager to restrict access to API proxies.

Walkthrough 5-1: Prepare an API for deployment using properties

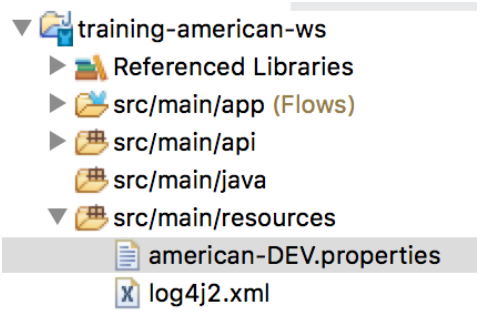
In this walkthrough, you introduce properties into your API implementation. You will:

- Create a properties file for an application.
- Create a Properties Placeholder global element to specify the properties file.
- Define and use Database connector properties.
- Specify a properties file dynamically.



Create a properties file

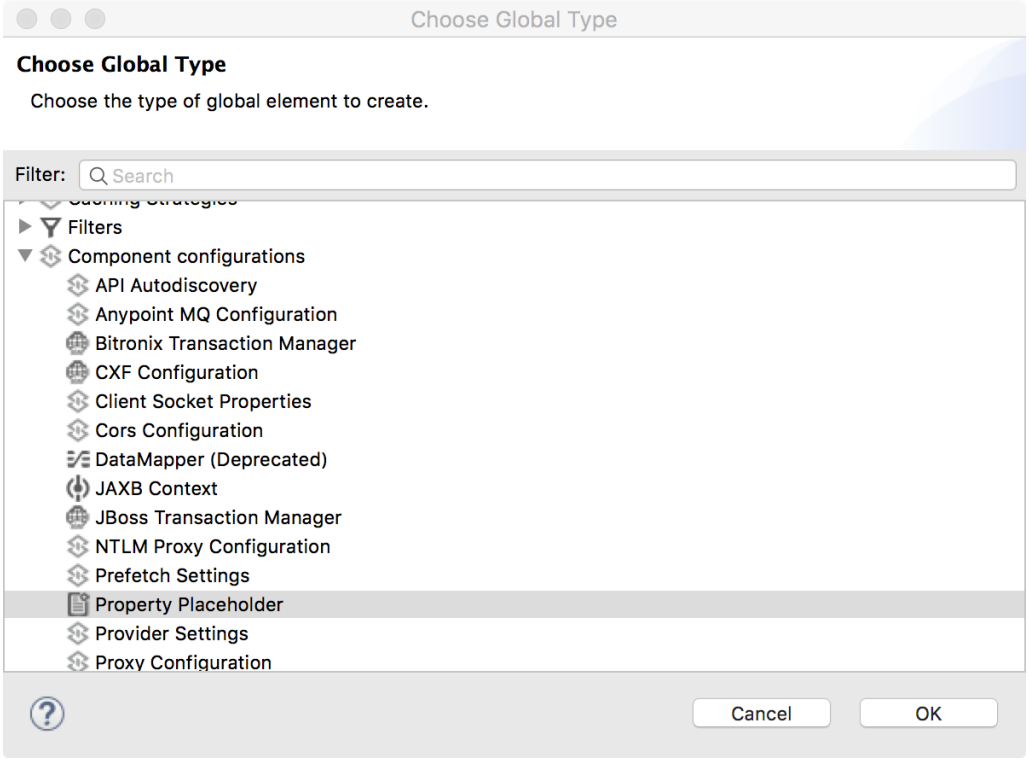
1. Right-click the src/main/resources folder in the Package Explorer and select New > File.
2. Set the file name to american-DEV.properties and click Finish.



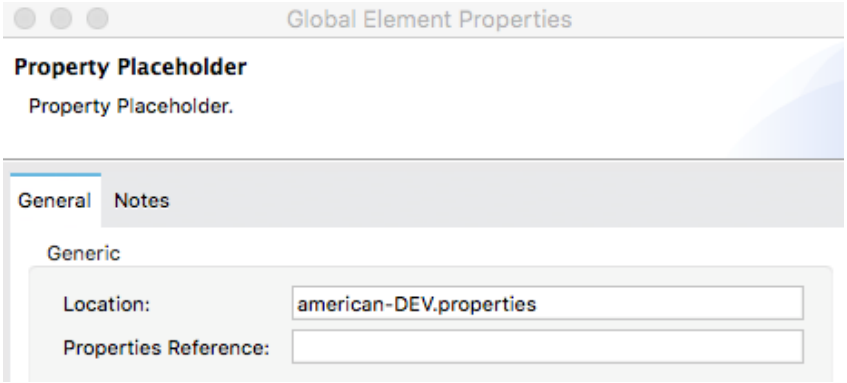
Create a Properties Placeholder global element

3. Return to interface.xml and go to its Global Elements view.

4. In the Global Elements view, click Create.
5. In the Choose Global Type dialog box, select Component configurations > Property Placeholder and click OK.



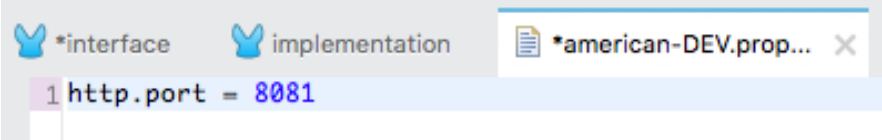
6. In the Global Element Properties dialog box, set the location to american-DEV.properties and click OK.



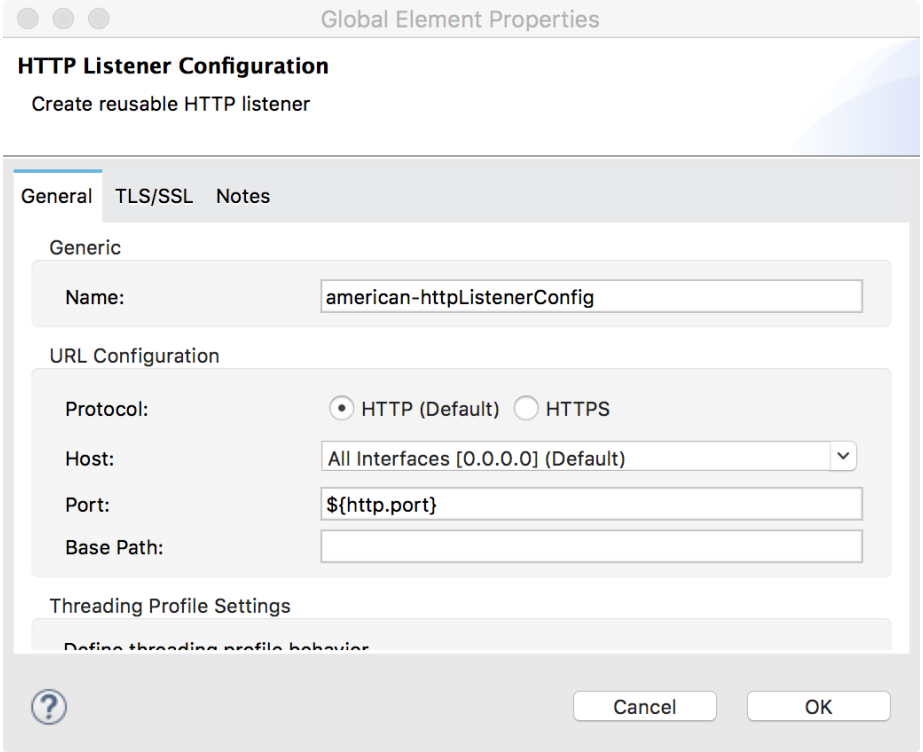
Parameterize the HTTP Listener port

7. Return to american-DEV.properties.

8. Create a property called http.port and set it to 8081.



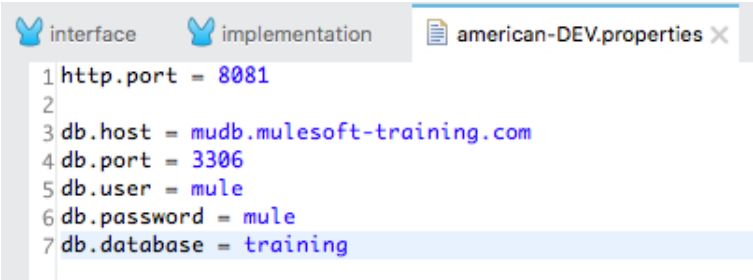
9. Save the file.
10. Return to the Global Elements view in interface.xml.
11. Double-click the HTTP Listener Configuration global element.
12. Change the port from 8081 to the application property, ${http.port}.
13. Click OK.



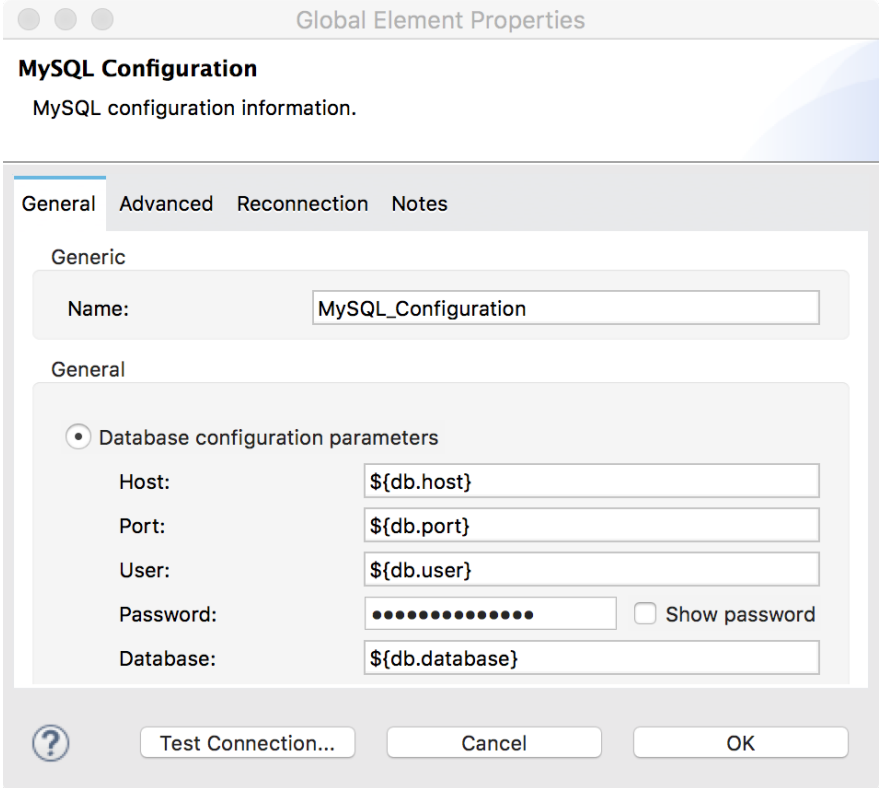
Parameterize the database credentials

14. Return to the course snippets.txt file and copy the database parameters (the five starting with db).

15. Return to american-DEV.properties and paste the values.



16. Save the file.
17. Return to the Global Elements view in implementation.xml.
18. Double-click the Database Configuration global element.
19. Change the values to use the application properties.



20. Click Test Connection and make sure it succeeds.

    Note: If your connection fails, click OK and then go back and make sure you saved american-DEV.properties.

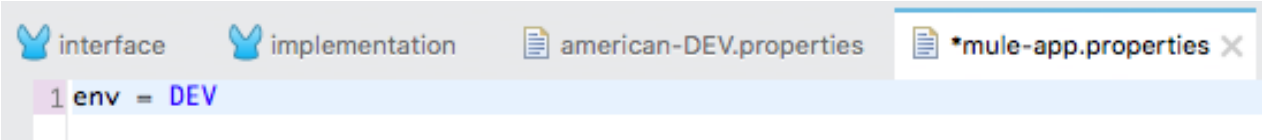
21. Click OK.
22. In the Global Element Properties dialog box, click OK.
23. Switch to the Message Flow view.

Test the application

24. Run the project.
25. In Postman, make a request http://localhost:8081/api/flights and confirm you still get data.

Define an environment property value in mule-app.properties

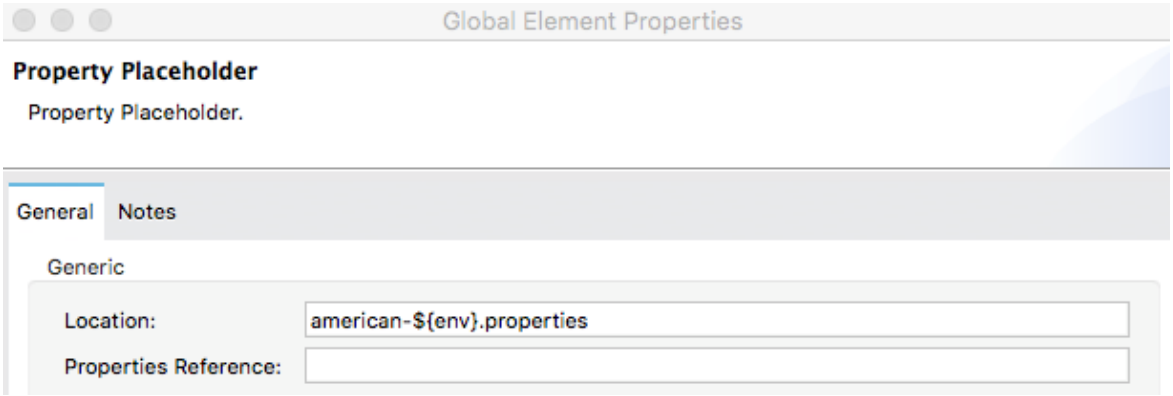
26. Return to Anypoint Studio and stop the project.
27. Open mule-app.properties located in the src/main/app folder.
28. Define a property called env and set it to DEV.



29. Save the file.

Use the environment property in the Property Placeholder

30. Return to the Global Elements view in interface.xml.
31. Double-click the Property Placeholder to edit it.
32. In the Global Element Properties dialog box, change the location to american-${env}.properties and click OK.



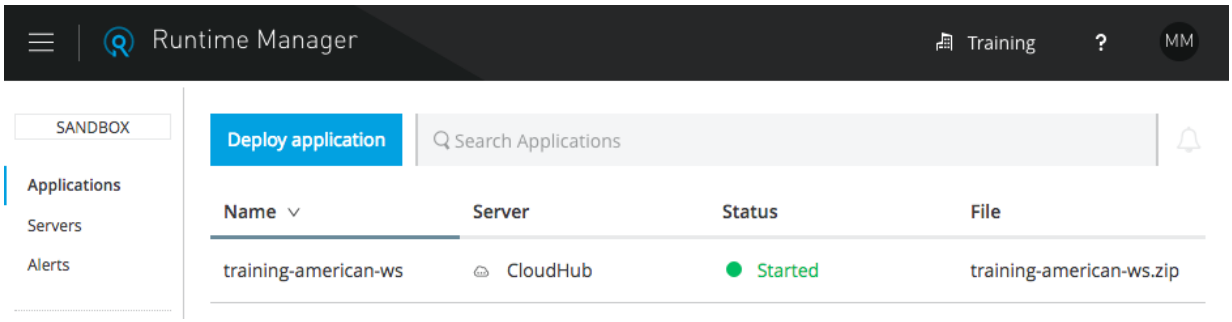
Test the application

33. Return to the Message Flow view in interface.xml.
34. Run the project.
35. In Postman, make a request http://localhost:8081/api/flights and confirm you still get data.
36. Return to Anypoint Studio and stop the project.

Walkthrough 5-2: Deploy an application to CloudHub

In this walkthrough, you deploy and run your application on CloudHub. You will:

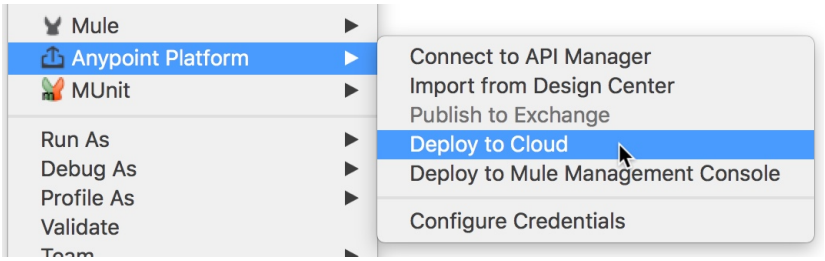
- Deploy an application from Anypoint Studio to CloudHub.
- Run the application on its new, hosted domain.
- Make calls to the web service.
- Update the API implementation deployed to CloudHub.



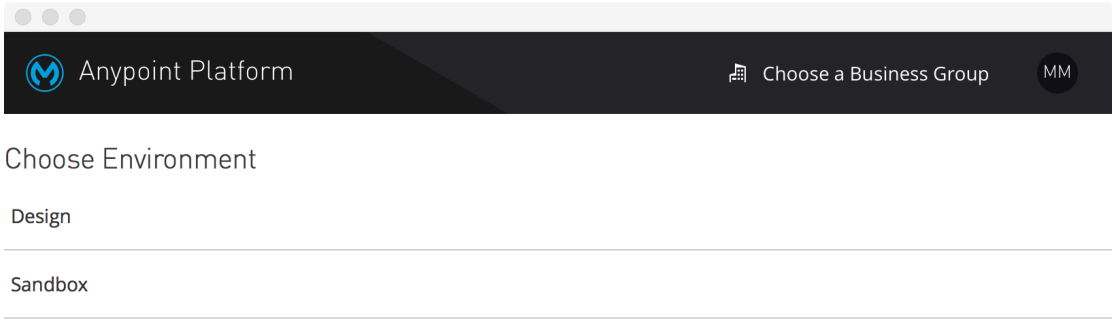
Note: If you do not have a working application at this point, import the training-american-ws-WT5.1 solution into Anypoint Studio and work with that project.

Deploy the application to CloudHub

1. Return to the training-american-ws project in Anypoint Studio.
2. In the Package Explorer, right-click the project and select Anypoint Platform > Deploy to Cloud.

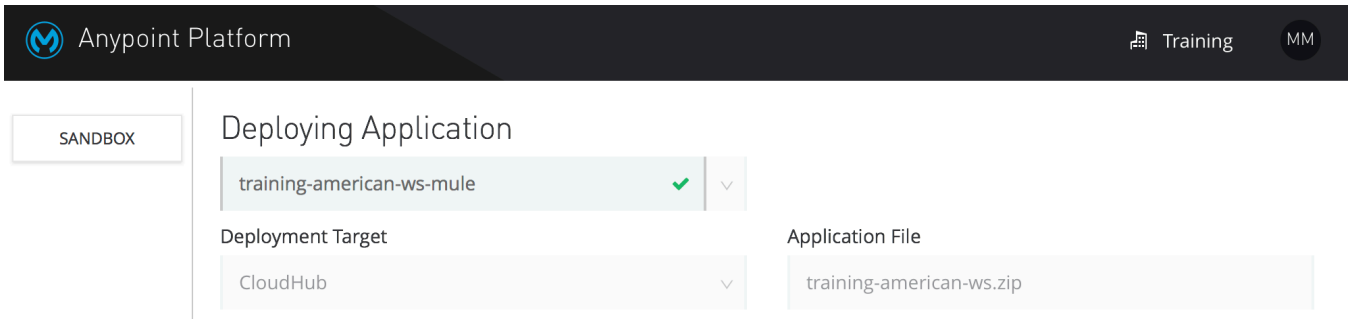


3. In the Choose Environment dialog box, select Sandbox.



4. At the top of the Anypoint Platform dialog box, set the application name to training-american-ws-*{your-lastname}* so it is a unique value.

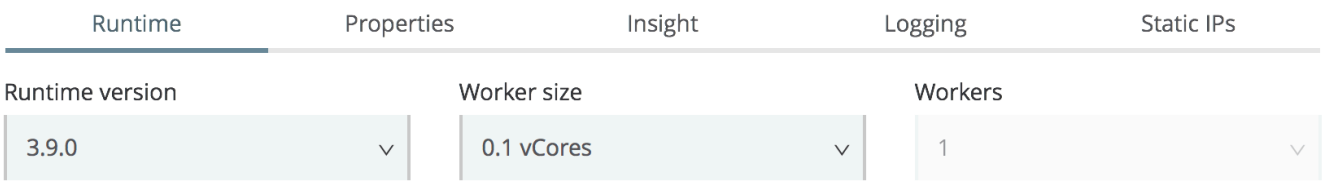
   Note: This name will be part of the URL used to access the application on CloudHub. It must be unique across all applications on CloudHub. The availability of the domain is instantly checked and you will get a green check mark if it is available.



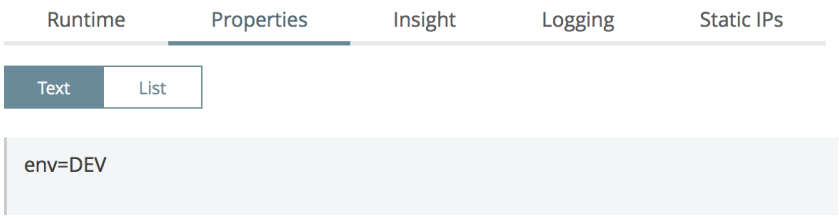
5. Make sure the runtime version is set to the version your project is using.

   Note: If you don't know what version it is using, look at the Package Explorer and find a library folder with the name of the server being used, like Mule Server 3.9.0 EE.

6. **Change the worker size to 0.1 vCores.**

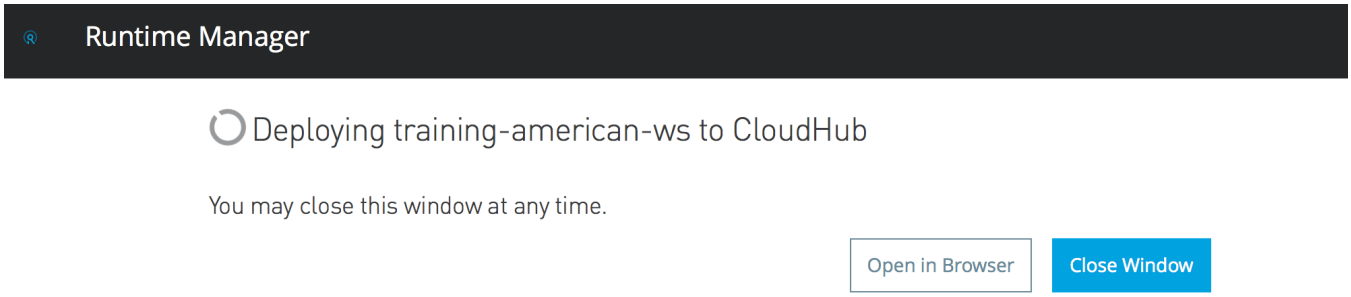


7. Click the Properties tab; you should see the env variable set to DEV.

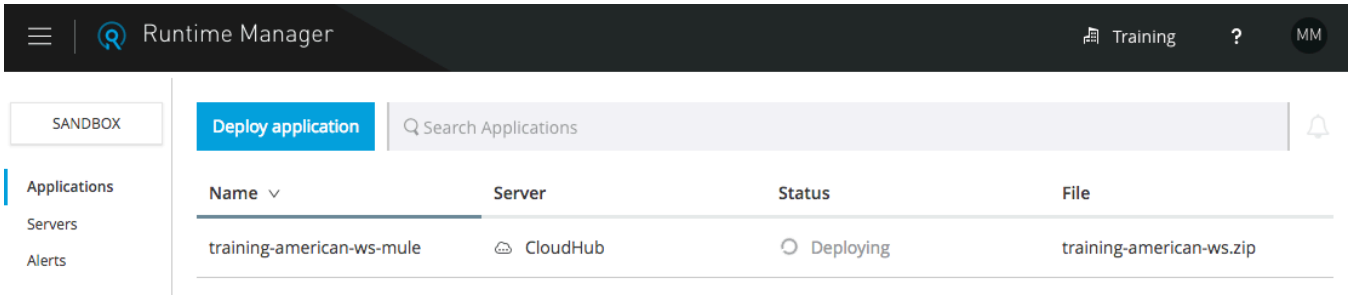


8. Click the Deploy Application button.

9. Click the Open in Browser button.

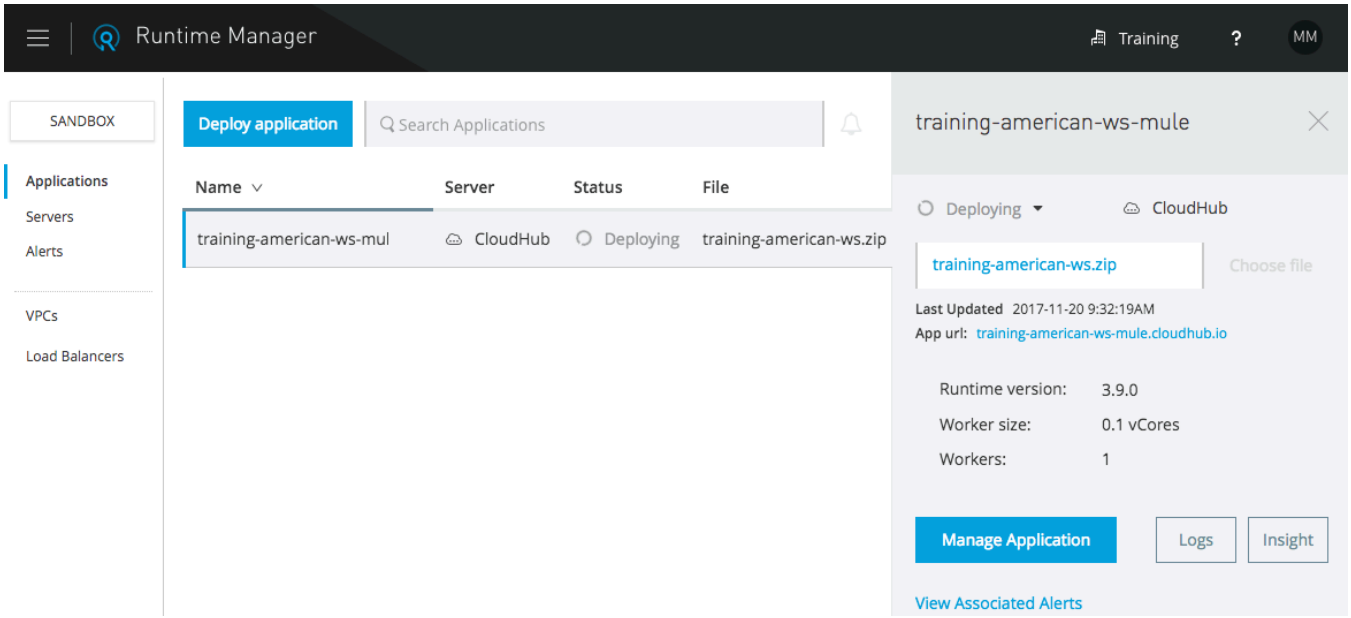


10. In the Anypoint Platform browser window that opens, locate the status of your deployment in the Runtime Manager.



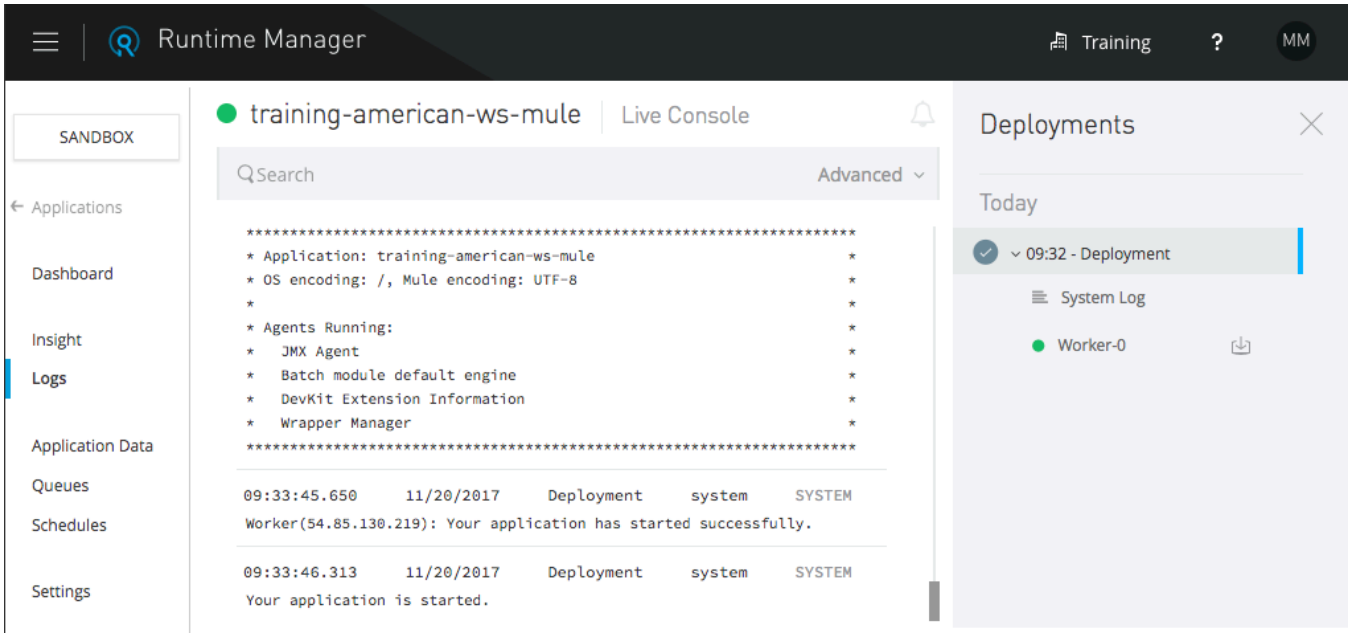
Watch the logs and wait for the application to start

11. Click in the row of the application (not on its name); you should see information about the application appear on the right side of the window.



12. Click the Logs button.

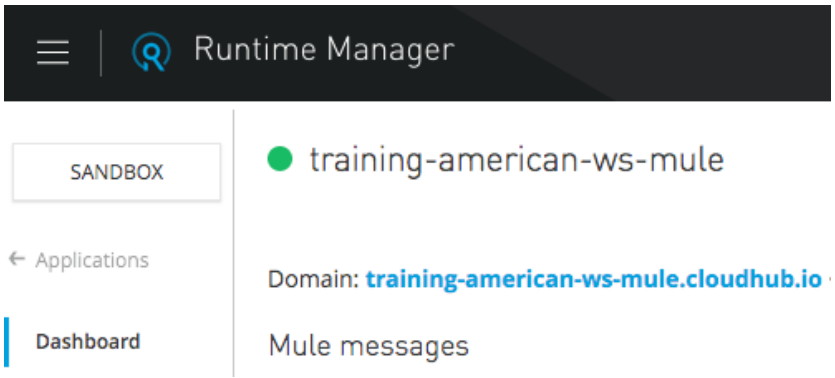
13. Watch the logs as the application is deployed.
14. Wait until the application starts (or fails to start).



Note: If your application did not successfully deploy, read the logs to help figure out why the application did not deploy. If you had errors when deploying, troubleshoot them, fix them, and then redeploy.

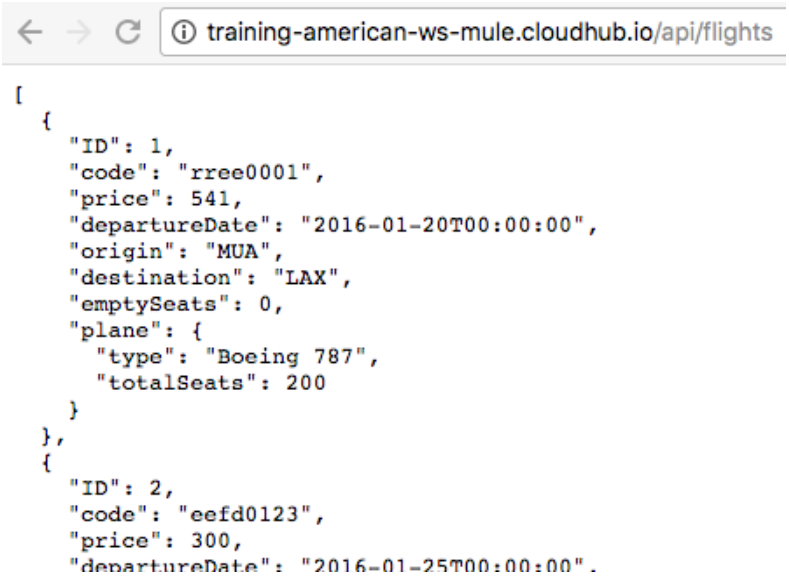
Test the application

15. In the left-side navigation, click Dashboard.
16. Locate the link for the application on its new domain: training-american-ws-*{lastname}*.cloudhub.io.



17. Click the link; a request will be made to that URL in an new browser tab and you should get a message that there is no listener for that endpoint.

18. Modify the path to http://training-american-ws-{*lastname*}.cloudhub.io/api/flights; you should see the flights data.



Note: If you are using the local Derby database, your application will not return results when deployed to CloudHub. You will update the application with a version using the MySQL database in the next section so it works.

19. Add a query parameter called destination to the URL and set it equal to SFO.
20. Send the request; you should still get all the flights.

Note: You did not add logic to the application to search for a particular destination. You will deploy an application with this additional functionality implemented next.

Note: If you wanted to test this API implementation from Exchange, you could by either 1) setting the baseUri in the API specification to the URL of the API implementation and then republishing the API specification or 2) clicking API instances in Exchange and adding a new API instance with this URL. Typically, however, you want to govern and manage access to an API. You will create a proxy to do this in the next walkthrough.

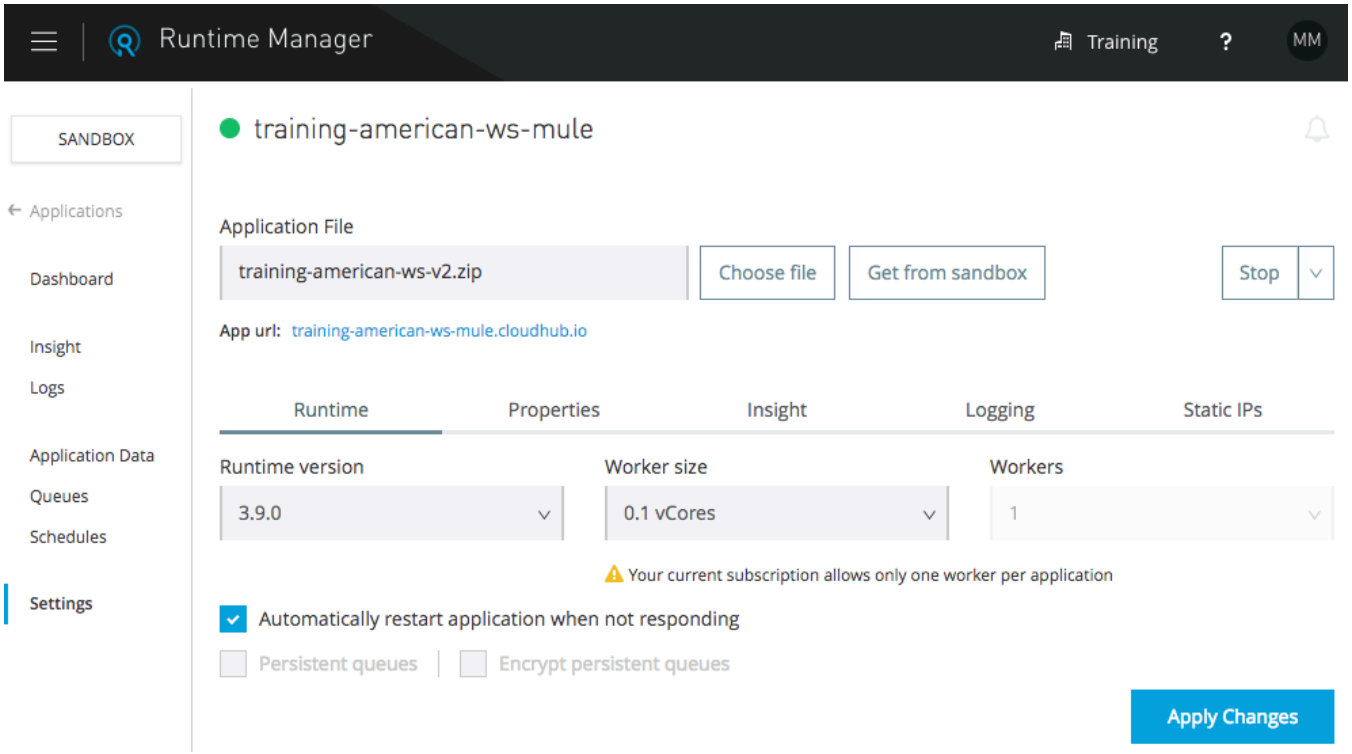
21. Leave this browser tab open.

Update the API implementation deployed to CloudHub

22. Return to the browser tab with Runtime Manager.
23. In the left-side navigation, click Settings for the training-american-ws-{*lastname*} application.
24. Click the Choose file button.
25. Browse to the resources folder in the course student files.
26. Select training-american-ws-v2.zip and click Open.

    Note: This updated version of the application adds functionality to return results for a particular destination. You will learn to do this later in the Development Fundamentals courses.

27. Click the Apply Changes button.



28. Wait until the application is uploaded and then redeploys successfully.

    Note: Because this can take some time for trial accounts, your instructor may move on with the next topic and then come back to test this later.

29. Close the browser tab with Runtime Manager.

Test the updated application

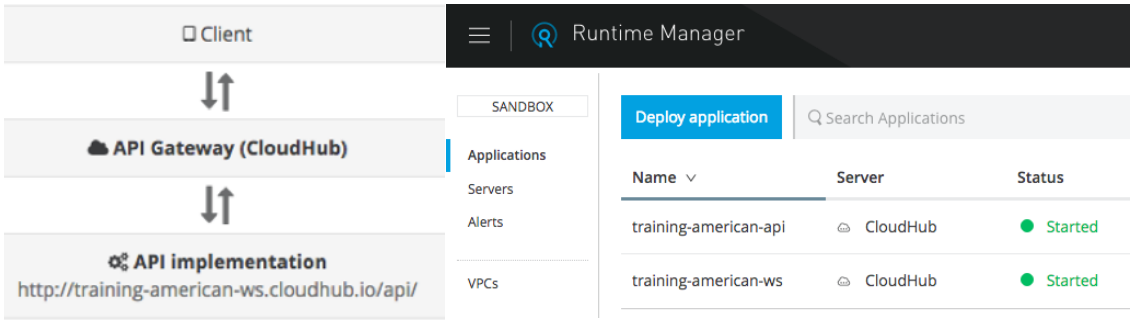
30. Return to the browser tab making a request to the API implementation on CloudHub with a destination of SFO and refresh it; you should now get only flights to SFO.

31. Close this browser tab.

Walkthrough 5-3: Create and deploy an API proxy

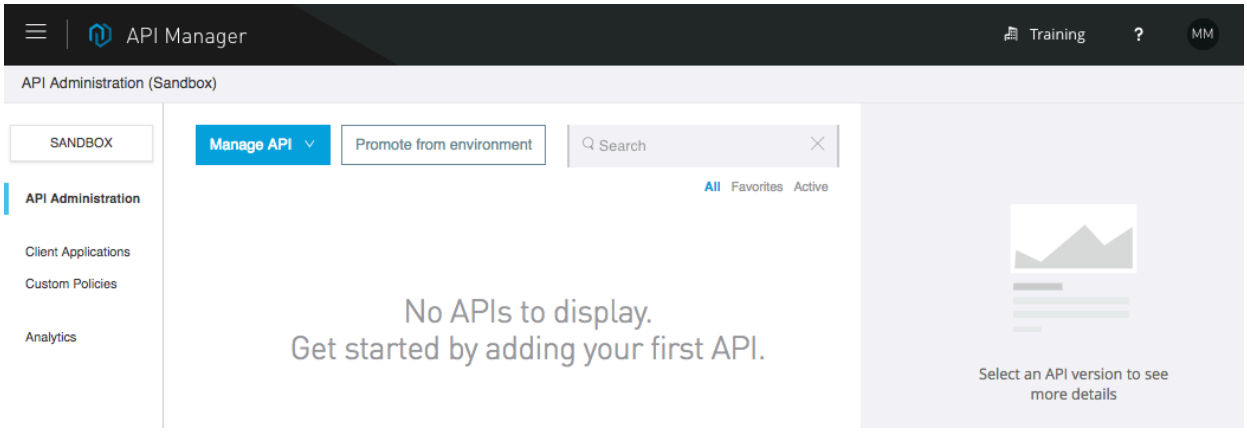
In this walkthrough, you create and deploy an API proxy for your API implementation on CloudHub. You will:

- Add an API to API Manager.
- Use API Manager to automatically create and deploy an API proxy application.
- Set a label and consumer endpoint for a proxy so requests can be made to it from Exchange.
- Make calls to the API proxy from API portals for both internal and external developers.
- View API request data in API Manager.

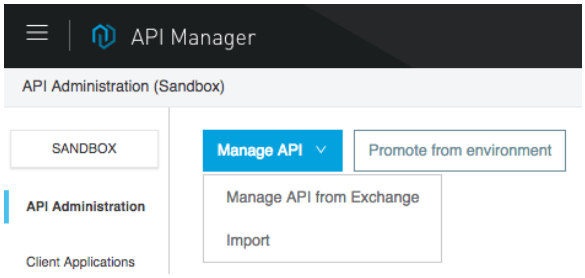


Create and deploy a proxy application

1. Return to the browser tab with Runtime Manager.
2. In the main menu, select API Manager; you should see no APIs listed.

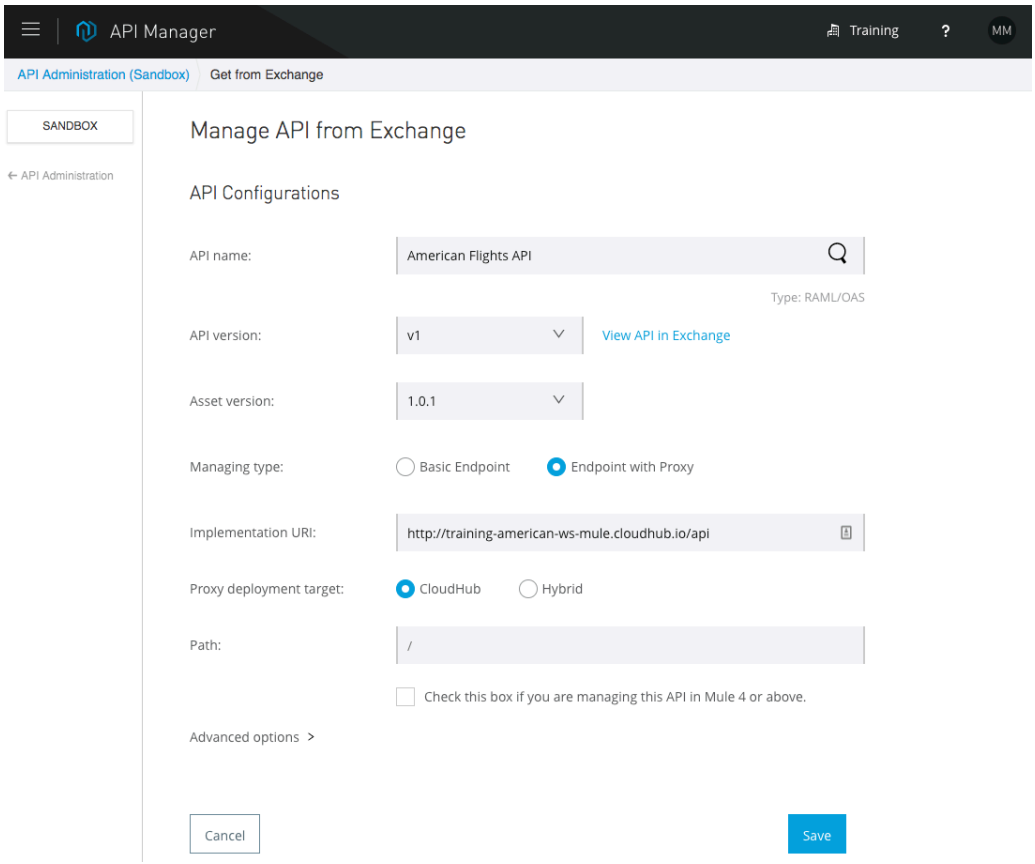


3. Click the Manage API button and select Manage API from Exchange.



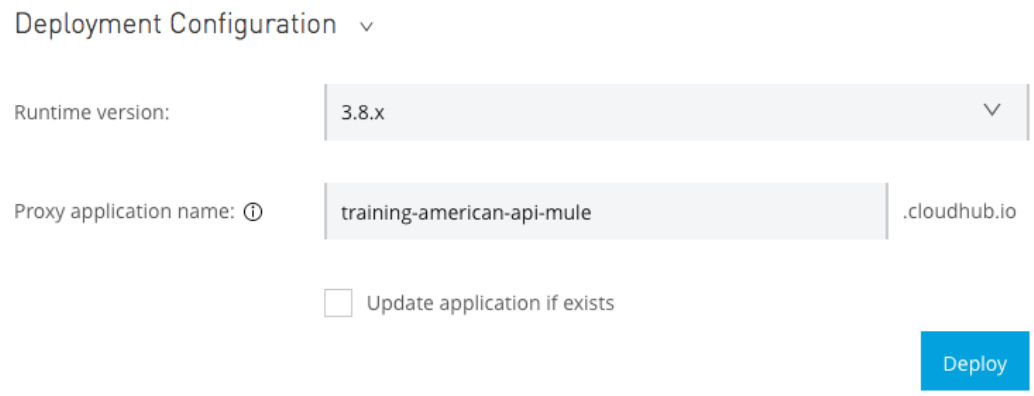
4. For API name, start typing American in the text field and then select your American Flights API in the drop-down menu that appears.
5. Set the rest of the fields to the following values:

   - API version: v1
   - Asset version: 1.0.1
   - Managing type: Endpoint with Proxy
   - Implementation URI: http://training-american-ws-{*lastname*}.cloudhub.io/api
   - Proxy deployment target: CloudHub

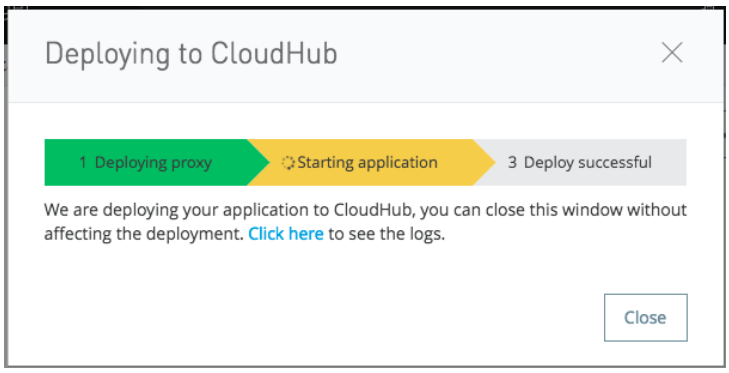


6. Click Save.

7. In the Deployment Configuration section, set the following values:
   - Runtime version: 3.8.x (or a later value)
   - Proxy application name: training-american-api-{*lastname*}

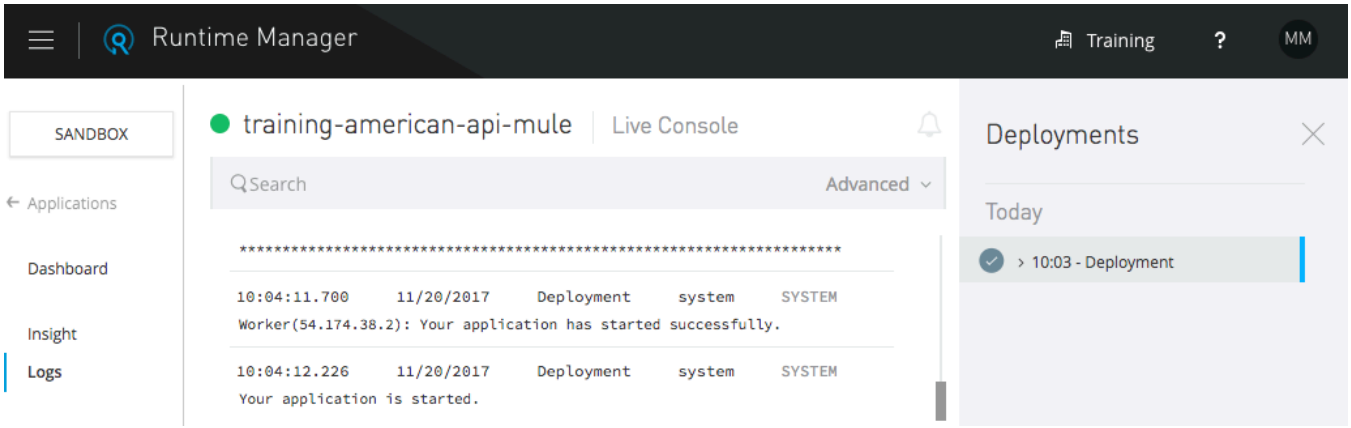


8. Click Deploy.
9. In the Deploying to CloudHub dialog box, click the Click here link to watch the logs.

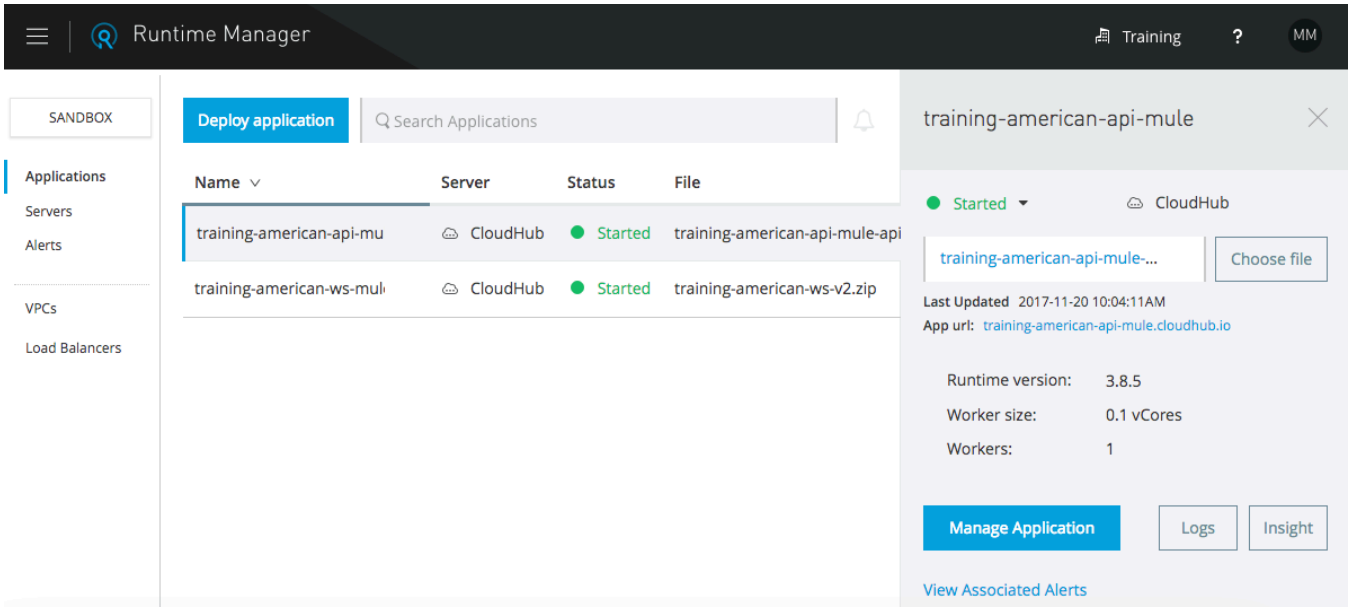


10. In the new browser tab that opens, watch the logs in Runtime Manager.
11. Wait until the proxy application starts.

    Note: If it does not successfully deploy, read the logs to help figure out why the application did not deploy. If you had errors when deploying, troubleshoot them, fix them, and then redeploy.



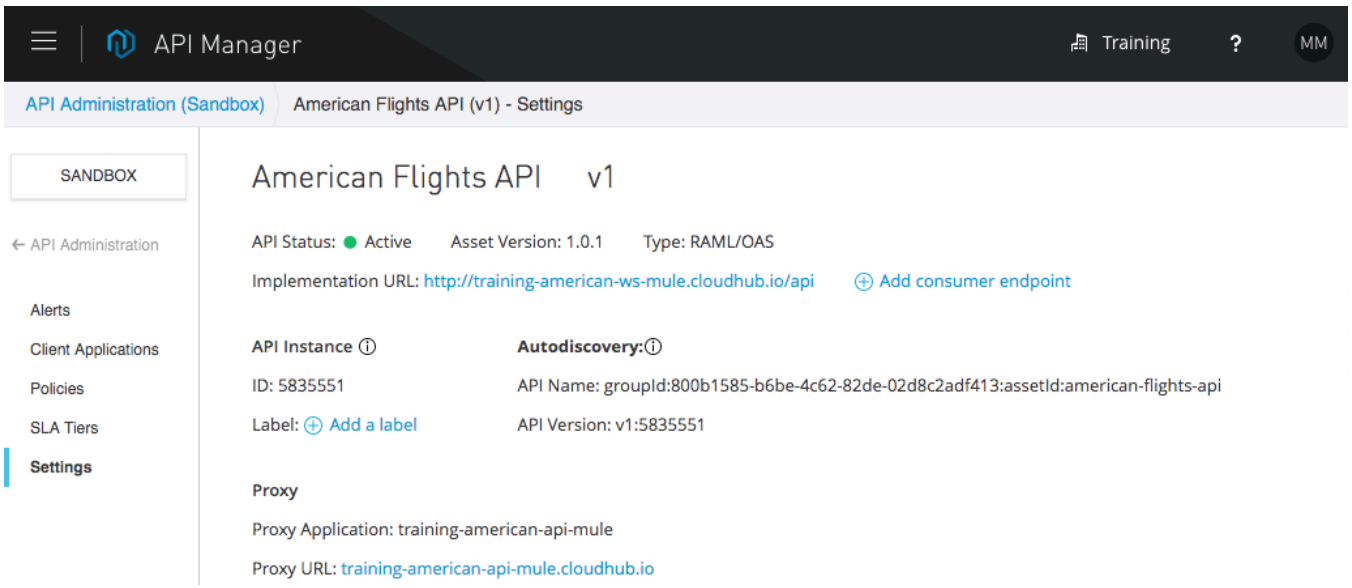
12. In the left-side navigation, click Applications; you should see the proxy application.
13. Click the row for the proxy and review its information in the right section of the window.



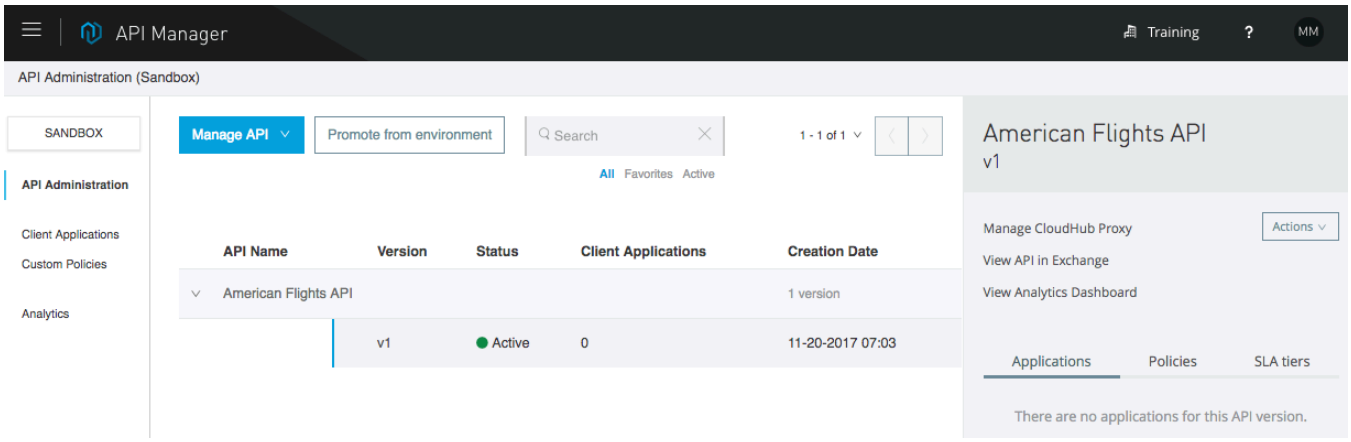
14. Close the browser tab.

View API details in API Manager

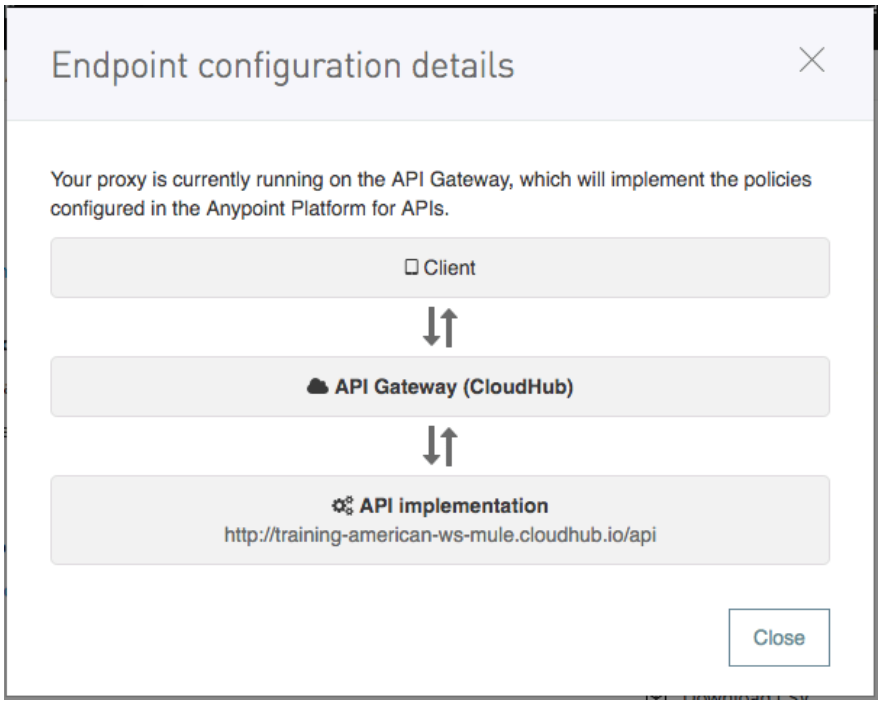
15. Return to the tab with API Manager and click the Close button in the Deploying to CloudHub dialog box.
16. Review the API proxy information at the top of the page.



17. In the left-side navigation, click the API Administration link; you should now see your American Flights API listed.
18. Click in the row for the v1 version – but not on the v1 link.
19. Review the API version info that appears on the right side of the window; you should see there are no policies, SLA tiers, or client applications.

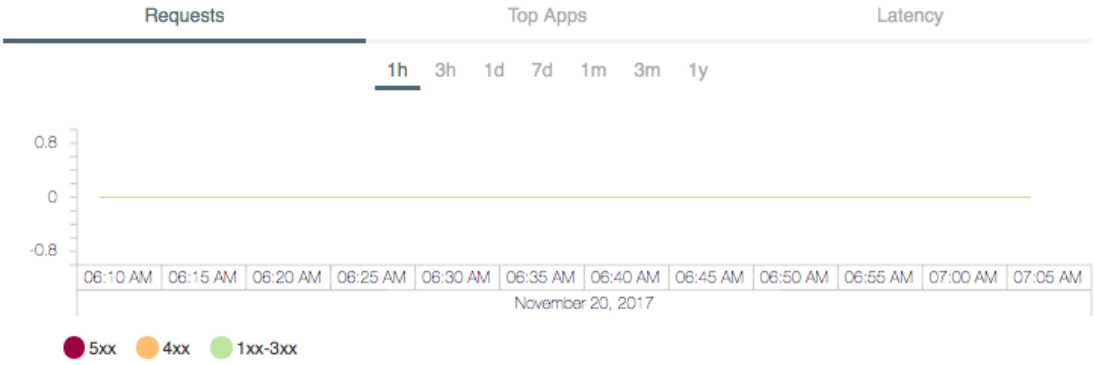


20. In the API list, click the v1 link for the API; you should be returned to the Settings page for the API.
21. Locate and review the links on the right side of the page.
22. Click the View configuration details link.



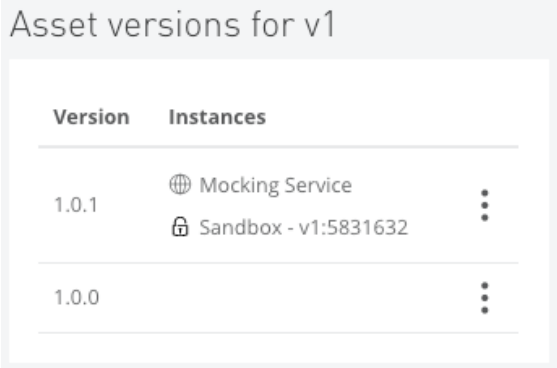
23. In the Endpoint configuration details dialog box, click Close.

24. On the Settings page, look at the requests graph; you should not see any API requests yet.



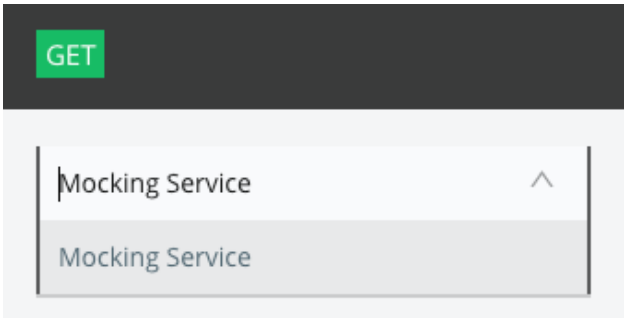
View the new API proxy instance in Exchange

25. Return to the browser tab with Exchange.
26. Return to the home page for your American Flights API.
27. Locate the API instances now associated with asset version 1.0.1; you should see the Mocking Service instance and now the new proxy.

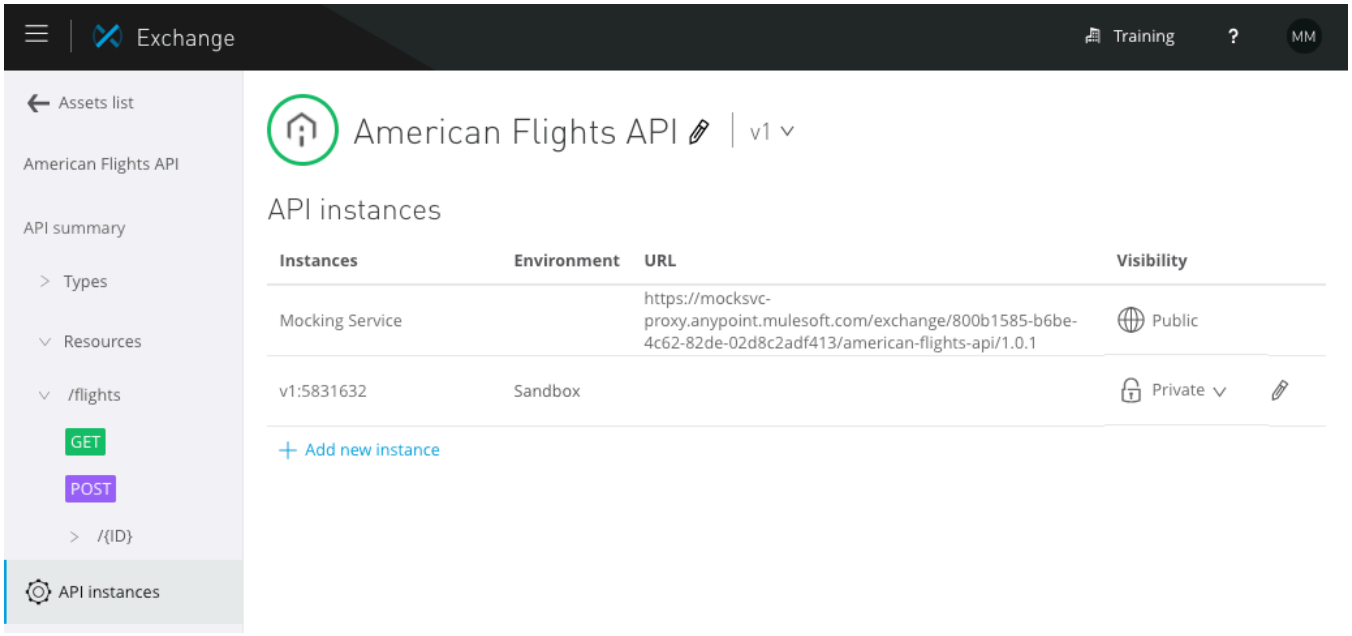


Note: You may need to refresh your browser page.

28. Click the GET method for the flights resource.
29. In the API console, click the drop-down arrow next to Mocking Service; you should NOT see the API proxy as a choice.

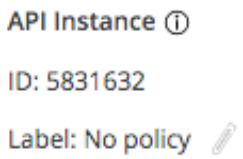


30. In the left-side navigation, click API instances; you should see that the new proxy instance does not have a URL.



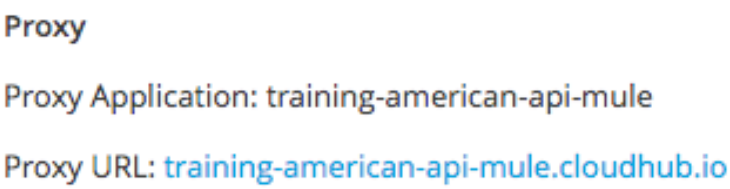
Set a friendly label for the API instance in API Manager

31. Return to the browser tab with API Manager.
32. On the Settings page for your American Flights API, click the Add a label link.
33. Set the label to No policy and press Enter/Return.



Set a consumer endpoint for the proxy in API Manager

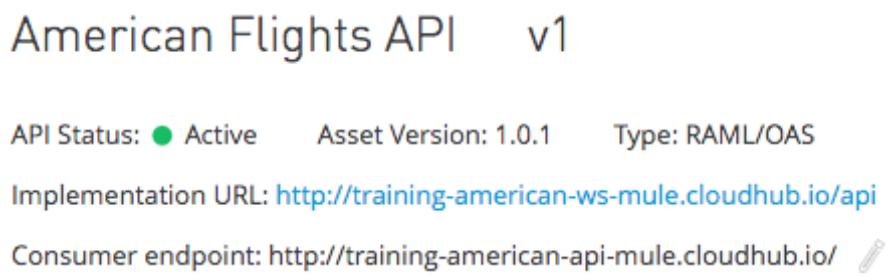
34. Locate the proxy URL.



35. Right-click it and copy the link address.

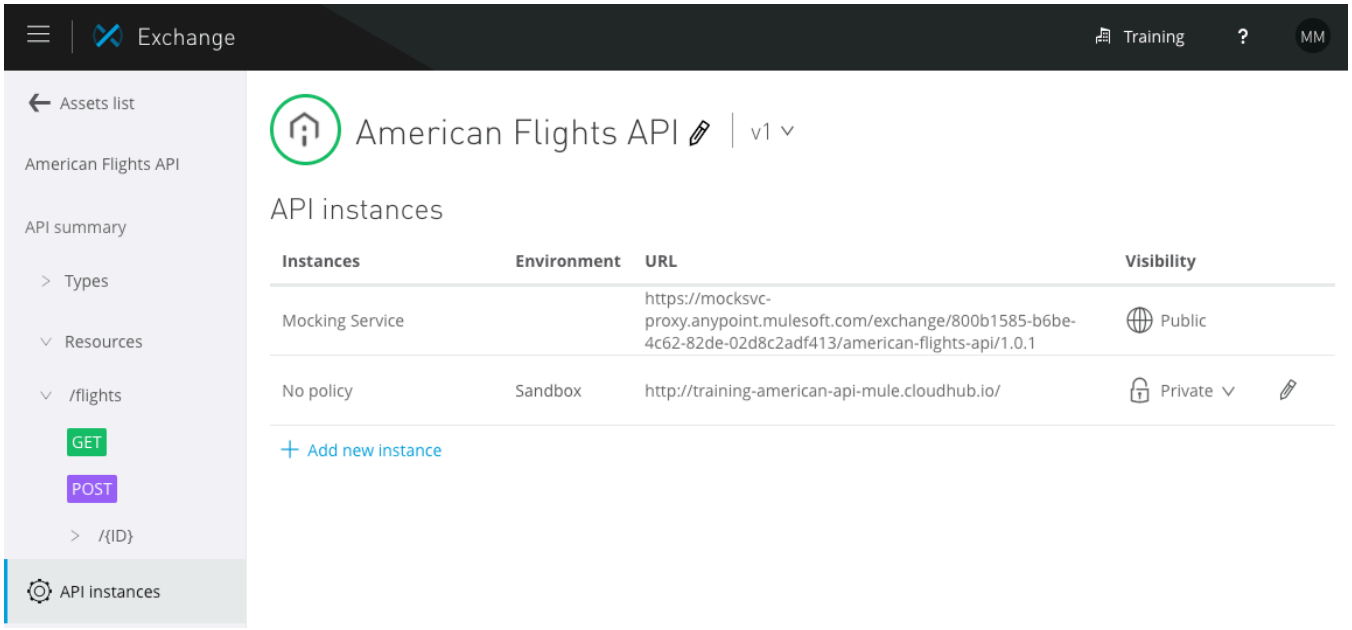
36. Click the Add consumer endpoint link.

37. Paste the value of the proxy URL.
38. Press Enter/Return.



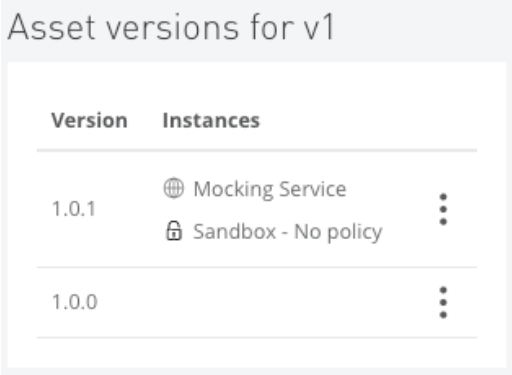
Make requests to the API proxy from Exchange

39. Return to the browser tab with Exchange.
40. Refresh the API instances page for your American Flights API; you should see the new label and the URL.

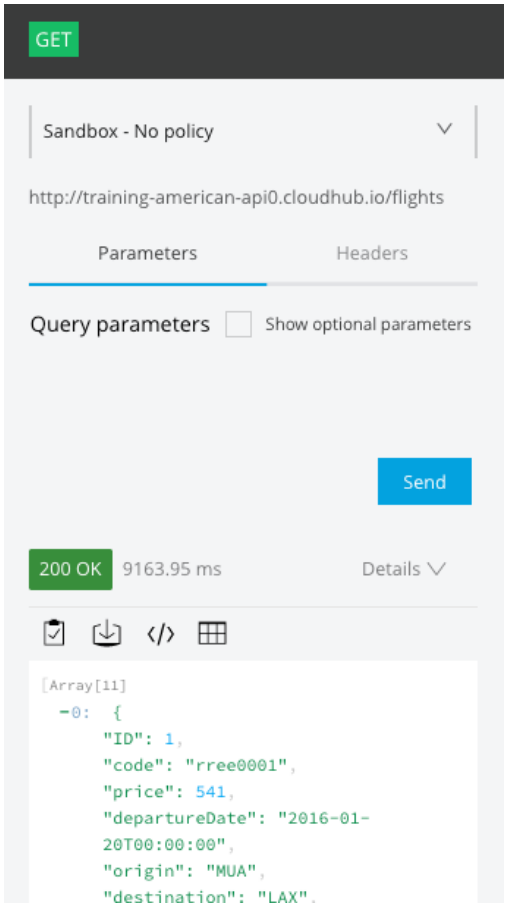


41. In the left-side navigation, click the name of the API to return to its main page.

42. Locate the API instances now associated asset version 1.0.1; you should see the new label for the API instance.



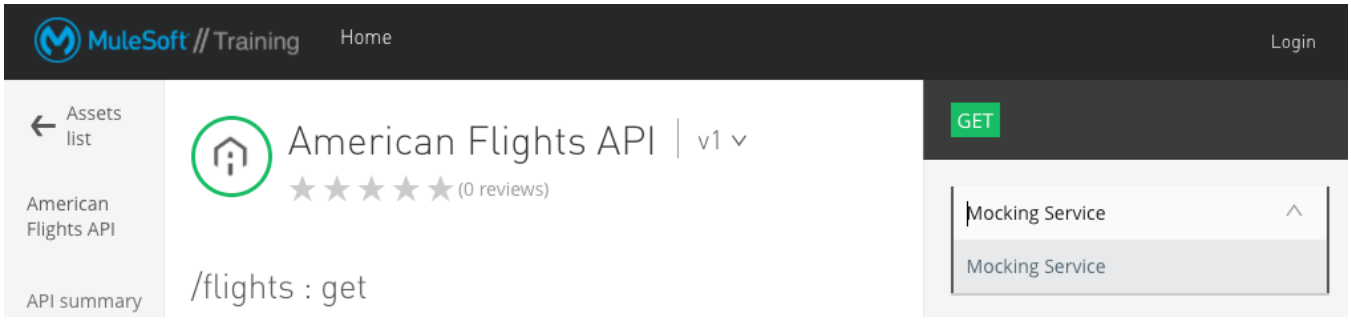
43. Click the GET method for the flights resource.
44. In the API console, click the drop-down arrow next to Mocking Service; you should now see your API proxy instance as a choice.
45. Select the Sandbox - No policy instance.
46. Click the Send button; you should now see the real data from the database, which contains multiple flights.



47. Make several more calls to this endpoint.
48. Make calls to different methods.

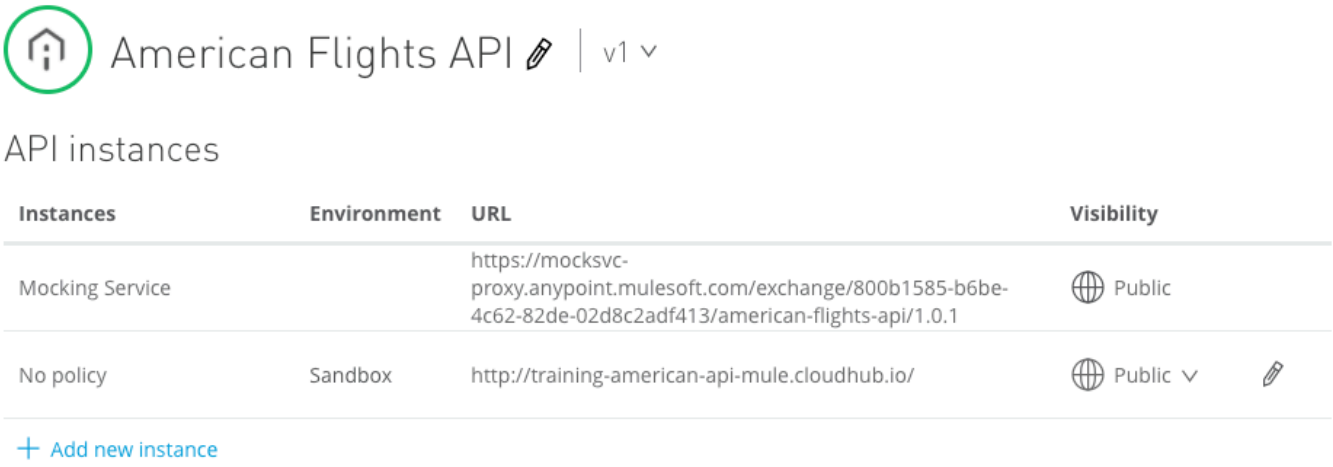
Make requests to the API proxy from the public portal

49. Return to the public portal in the private/incognito window.
50. Click the GET method for the flights resource.
51. In the API reference section, click the drop-down arrow next to Mocking Service; you should NOT see your API proxy instance as a choice.



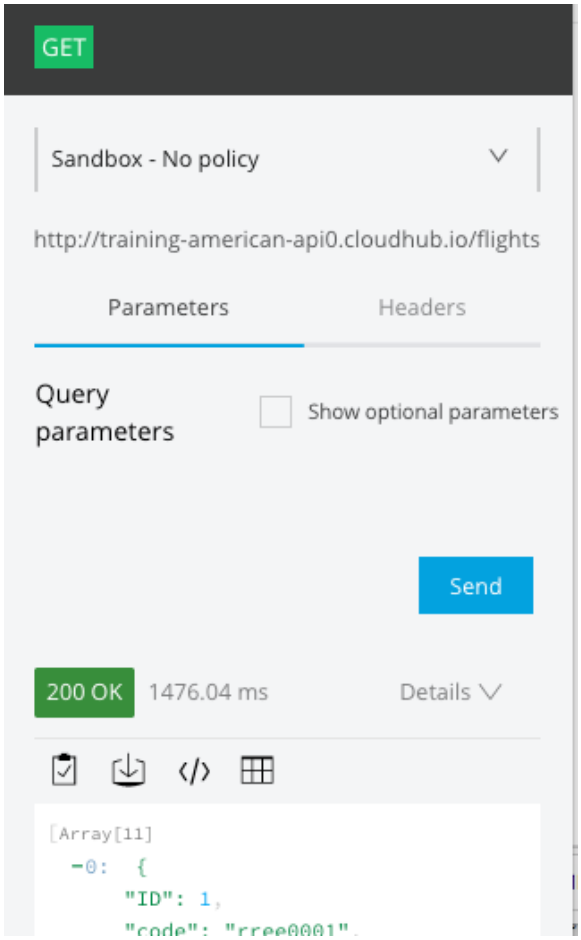
Make an API instance visible in the public portal

52. Return to the browser with Exchange.
53. In the left-side navigation, click API instances.
54. Change the visibility of the No policy instance from private to public.



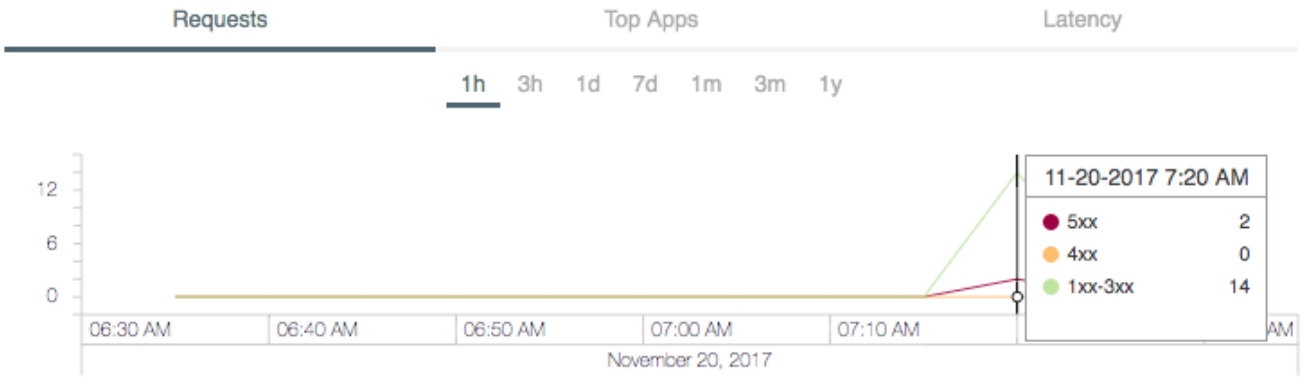
55. Return to the public portal in the private/incognito window.
56. Refresh the page.

57. In the API console, change the API instance from Mocking Service to Sandbox - No policy.
58. Click Send; you should get data.

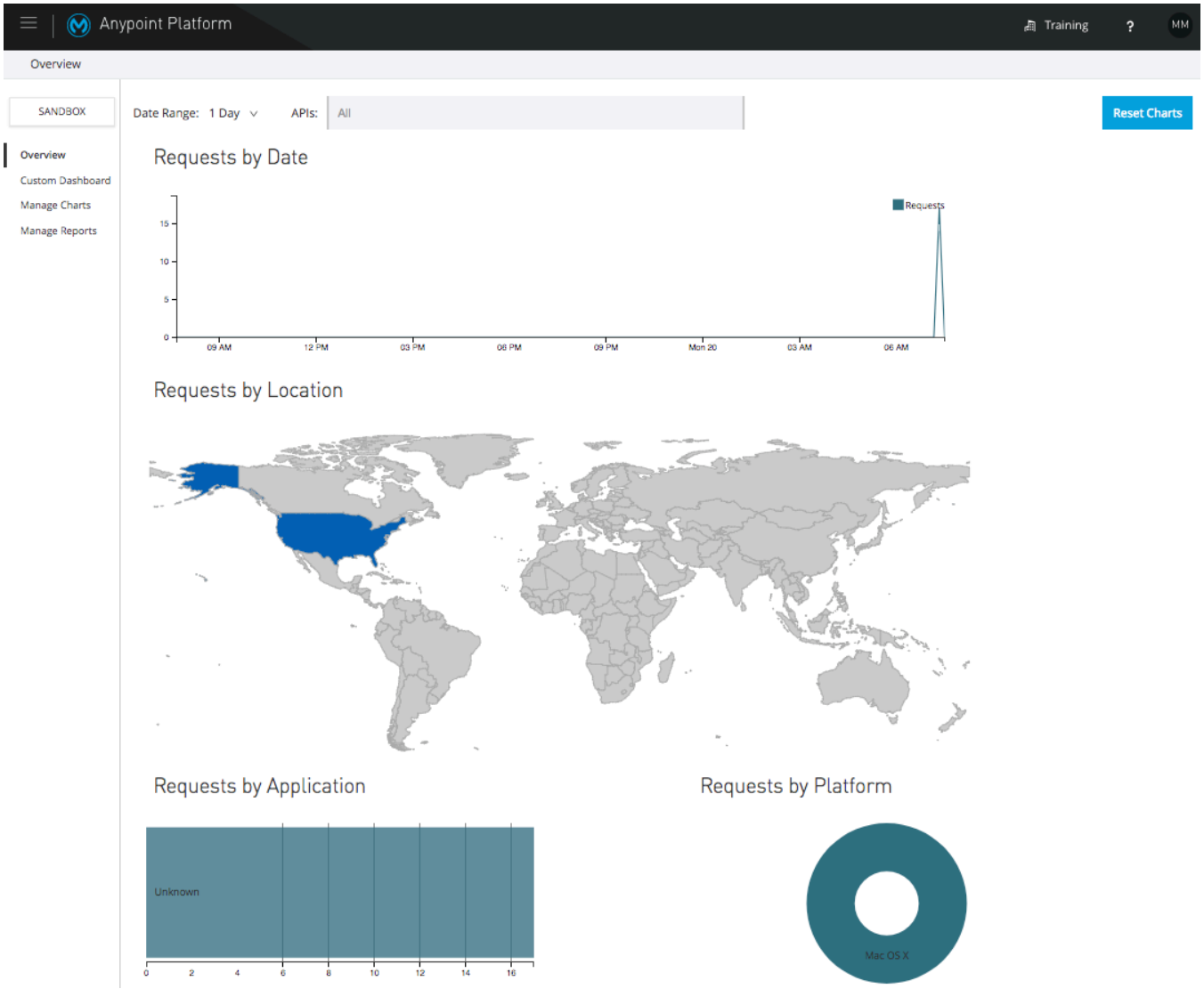


Look at the API request data

59. Return to the browser tab with API Manager.
60. Refresh the Settings page for you American Flights API.
61. Look at the Request chart again; you should now see data for some API calls.



62. Click the View Analytics Dashboard link located in the upper-right corner.
63. Review the data in the dashboard.

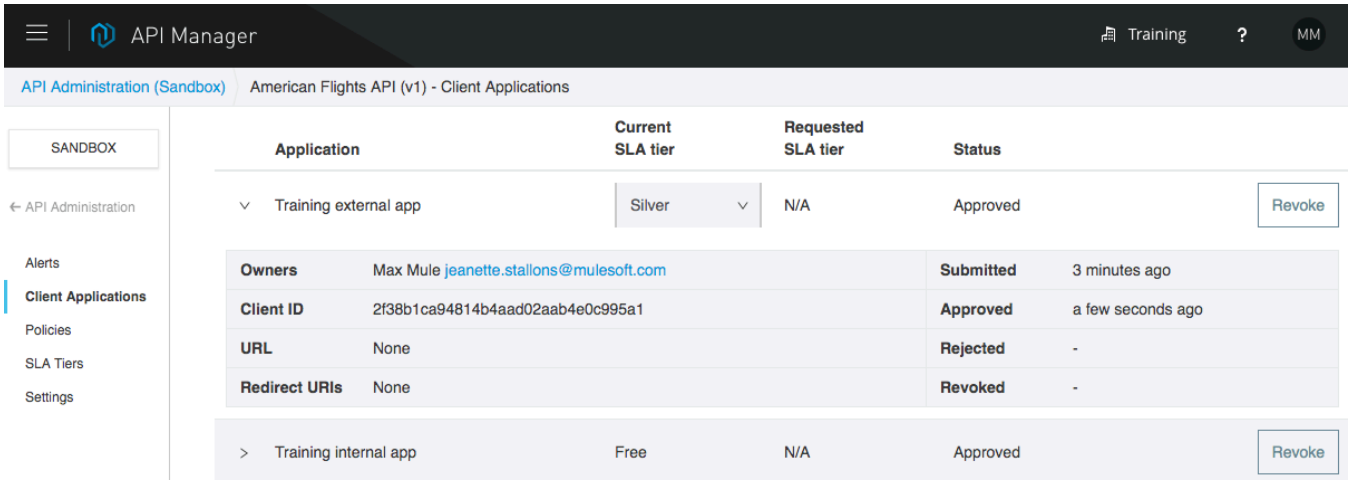


64. Close the browser tab.

Walkthrough 5-4: Restrict API access with policies and SLAs

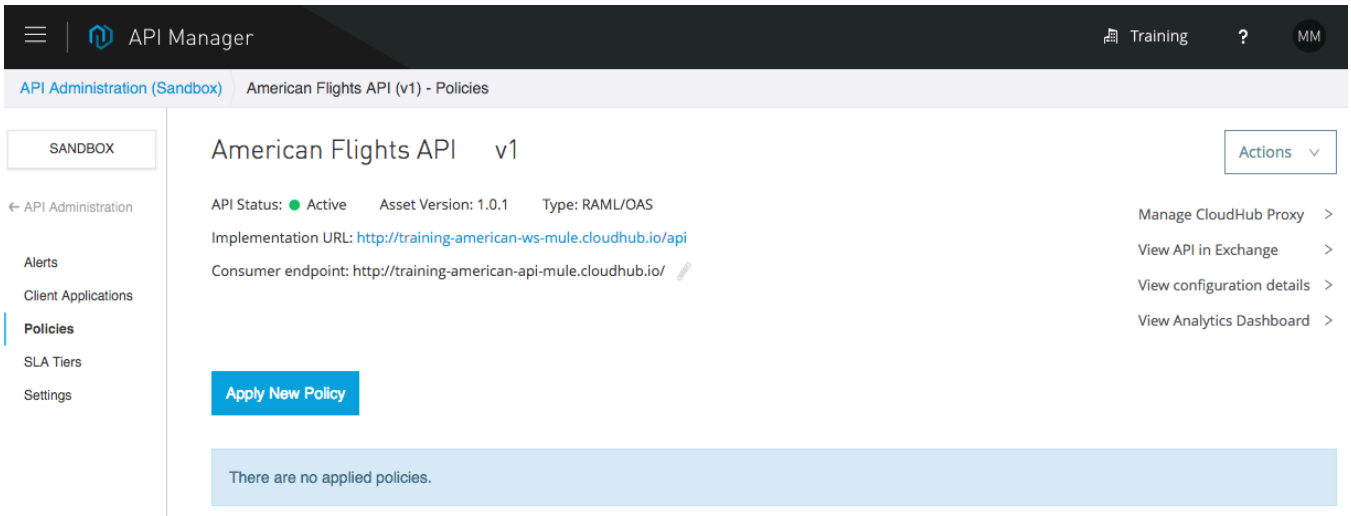
In this walkthrough, you govern access to the API proxy. You will:

- Add and test a rate limiting policy.
- Add SLA tiers, one with manual approval required.
- Add and test a rate limiting SLA policy.
- Request application access to SLA tiers from private and public API portals.
- Approve application requests to SLA tiers in API Manager.



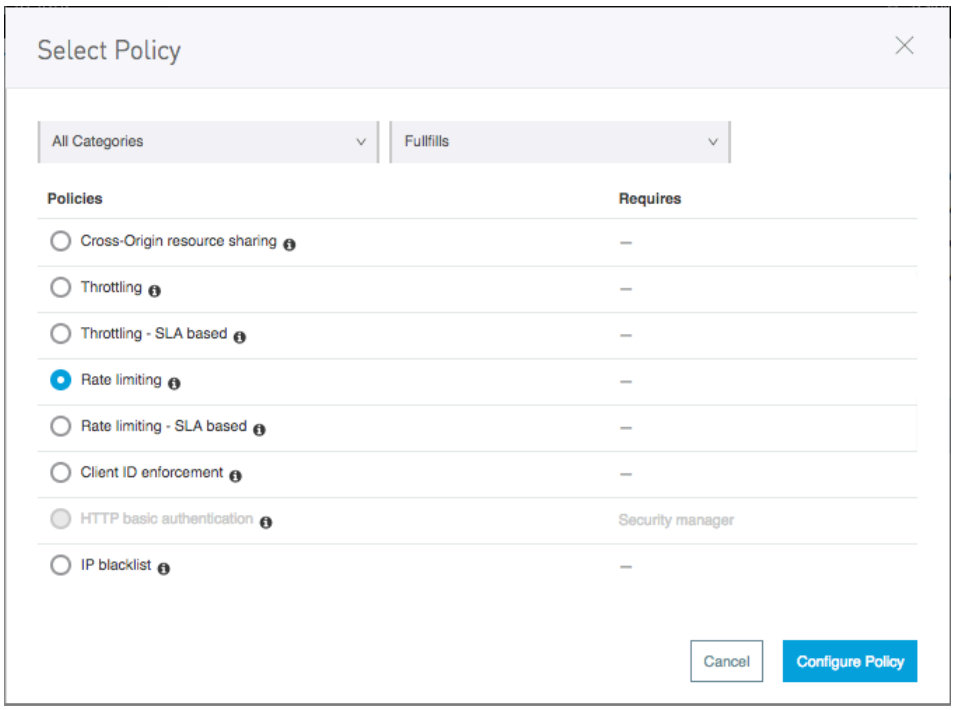
Create a rate limiting policy

1. Return to the Settings page for your American Flights API in Anypoint Manager.
2. In the left-side navigation, click the Policies link.



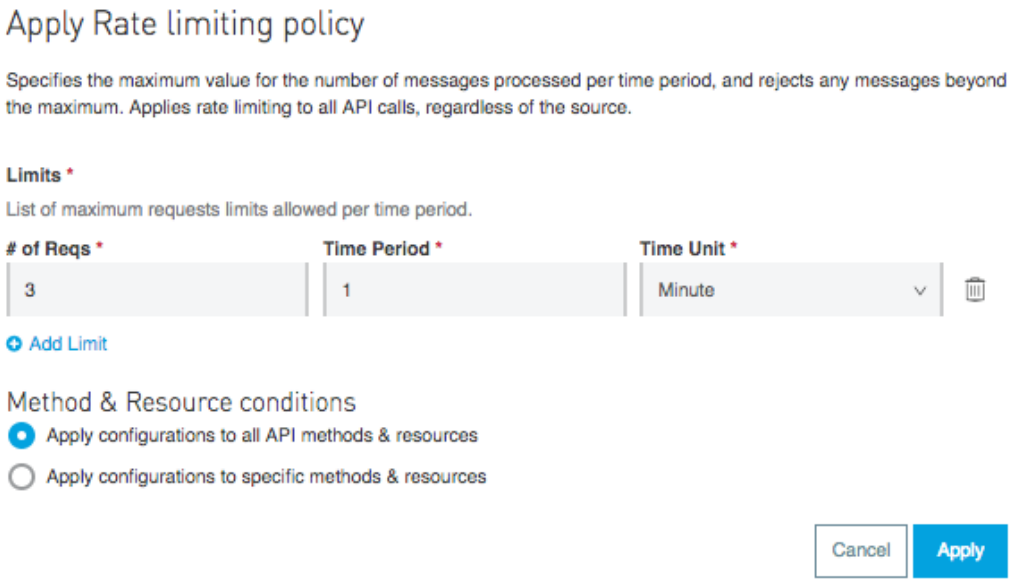
3. Click the Apply New Policy button.

4. In the Select Policy dialog box, select Rate limiting.

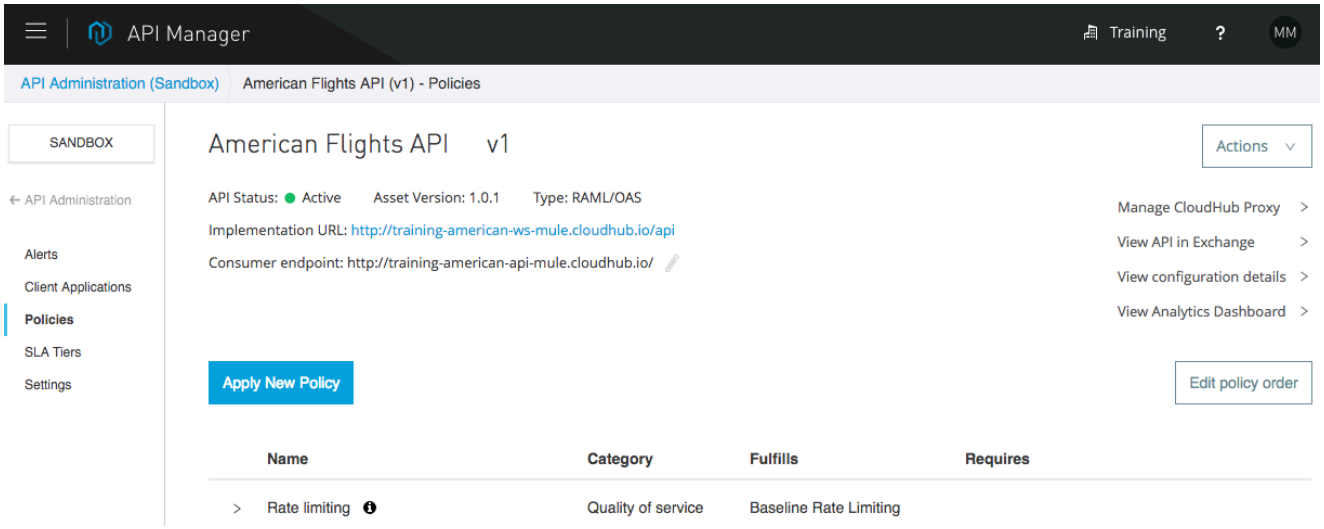


5. Click Configure Policy.
6. On the Apply Rate limiting policy page, set the following values and click Apply:

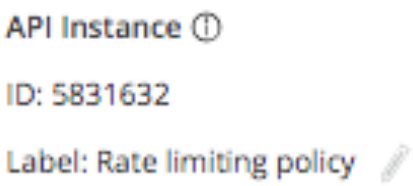
   - # of Reqs: 3
   - Time Period: 1
   - Time Unit: Minute
   - Methods & Resource conditions: Apply configurations to all API methods & resources



7. Click Apply; you should see the policy listed for your API.



8. In the left-side navigation, click Settings.
9. Change the API instance label to Rate limiting policy.

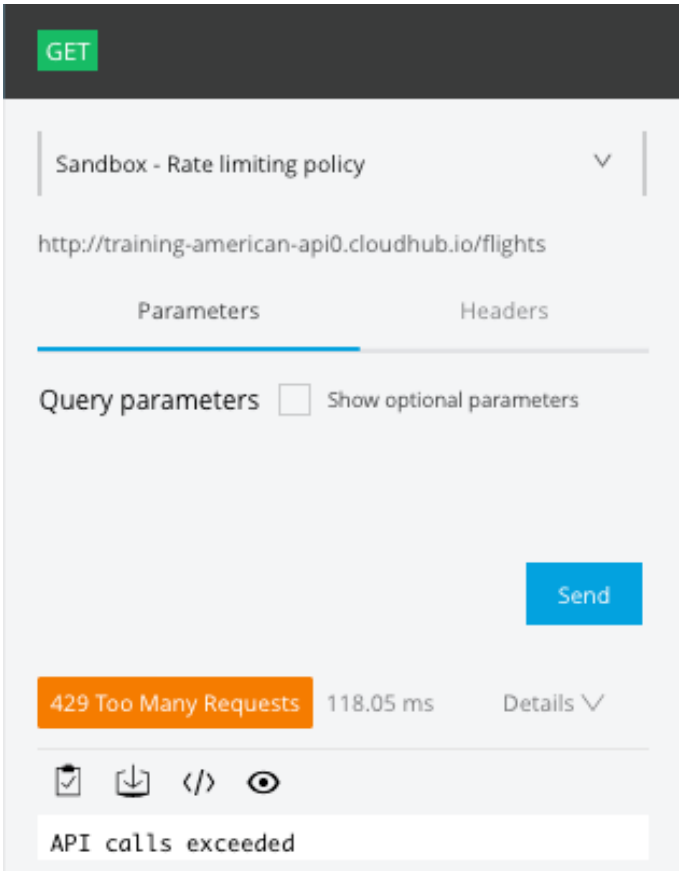


Test the new rate limiting policy

10. Return to the browser tab with your American Flights API in Exchange.
11. Return to the page with the API console for the flights:/GET resource.
12. Select the Sandbox – Rate limiting policy API instance.

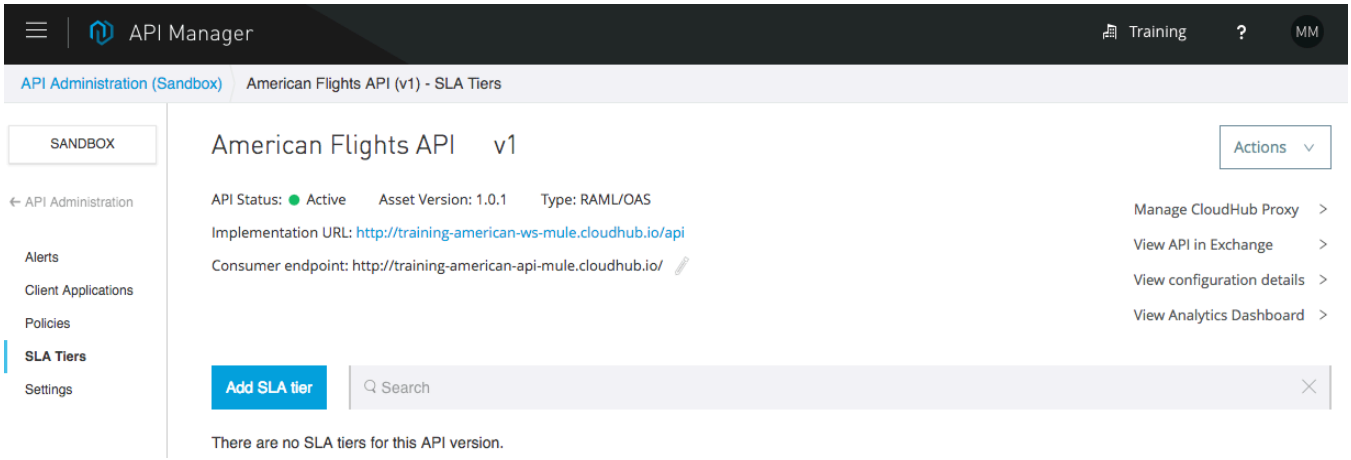
    Note: You may need to refresh the page to see the new label for the API instance.

13. Press Send until you get a 429 Too Many Requests response.



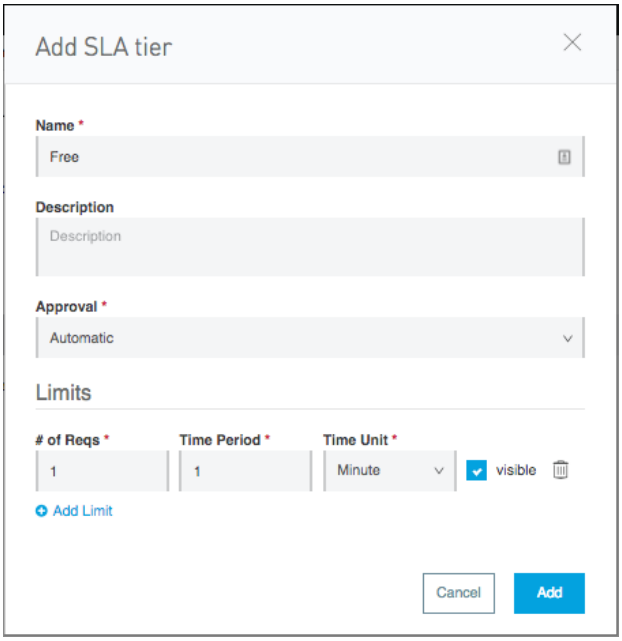
Create SLA tiers

14. Return to the browser tab with your American Flights API in API Manager.
15. In the left-side navigation, click the SLA Tiers link.
16. Click the Add SLA tier button.



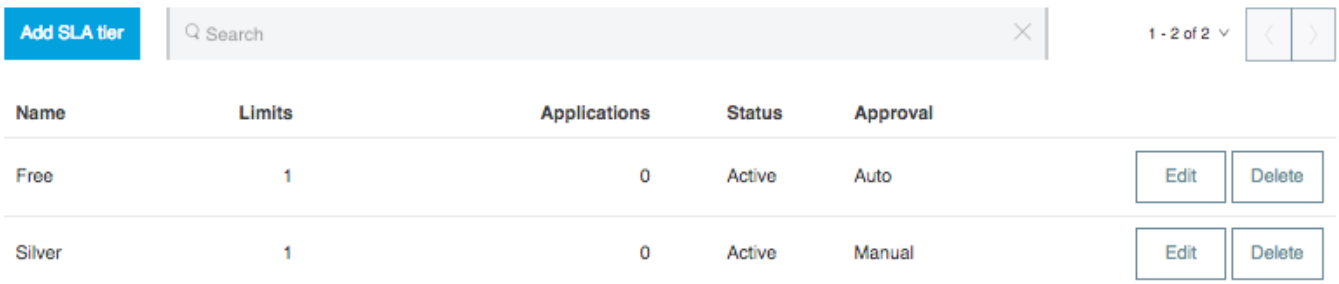
17. In the Add SLA tier dialog box, set the following values:

- Name: Free
- Approval: Automatic
- # of Reqs: 1
- Time Period: 1
- Time Unit: Minute



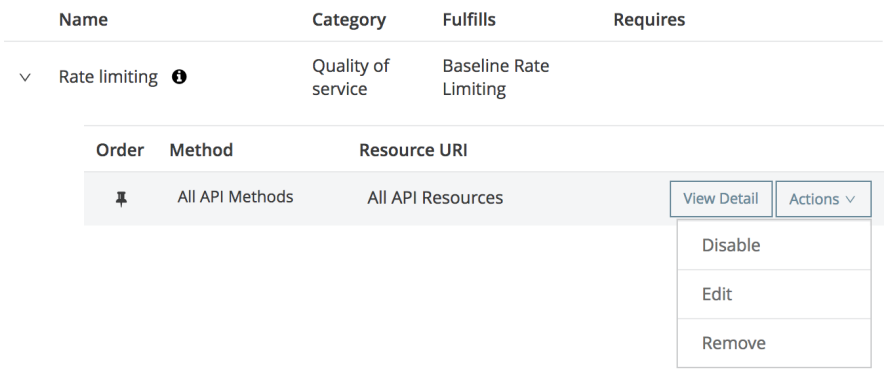
18. Click the Add button.
19. Create a second SLA tier with the following values:

- Name: Silver
- Approval: Manual
- # of Reqs: 1
- Time Period: 1
- Time Unit: Second

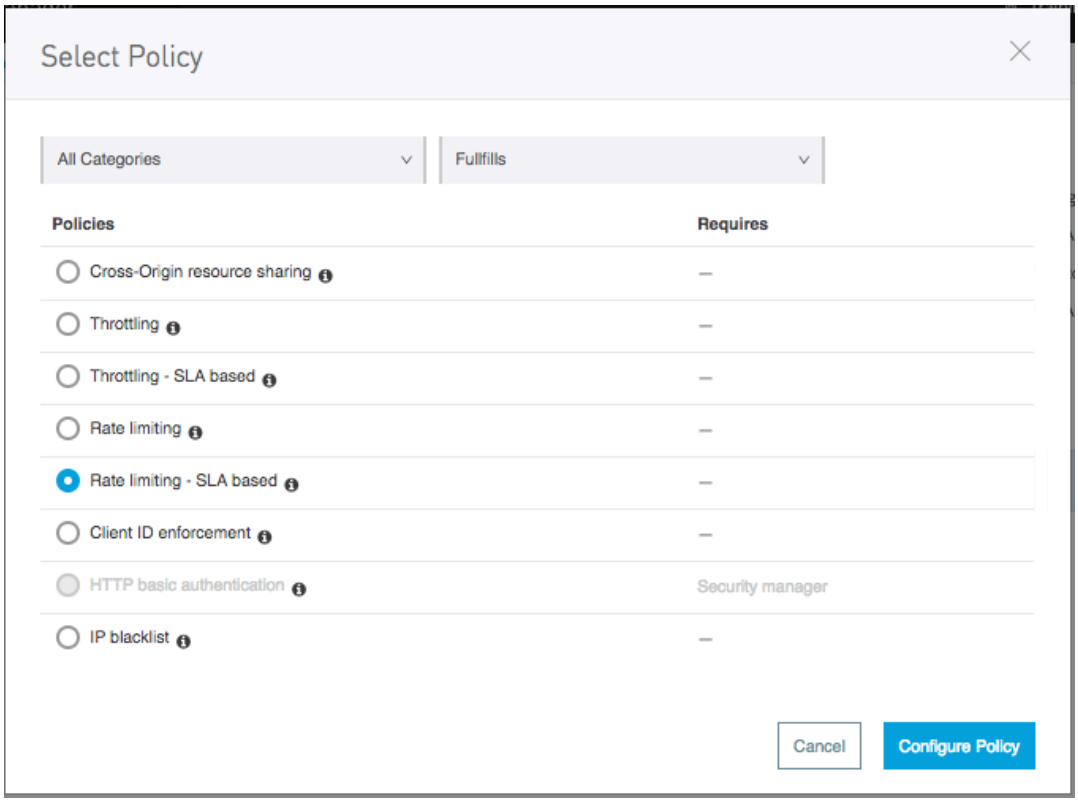


Change the policy to rate limiting – SLA based

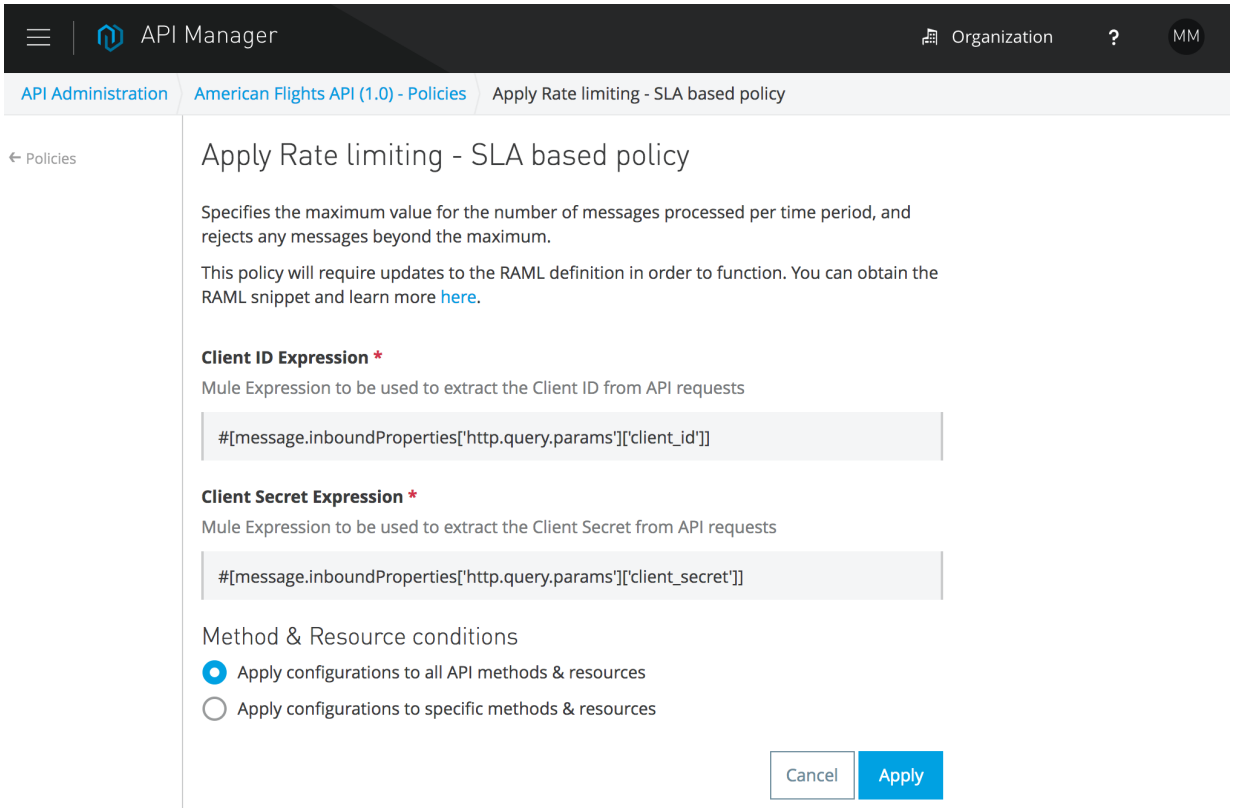
20. In the left-side navigation, click the Policies link.
21. Expand the Rate limiting policy.
22. Click the Actions button and select Remove.



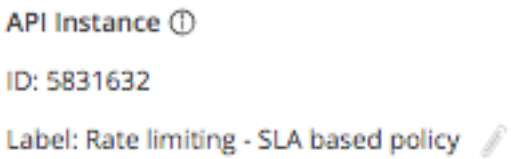
23. In the Remove policy dialog box, click Remove.
24. Click the Apply New Policy button.
25. In the Select Policy dialog box, select Rate limiting - SLA based.
26. Click Configure Policy.



27. On the Apply Rate limiting – SLA based policy page, look at the expressions and see that a client ID and secret need to be sent with API requests as query parameters.



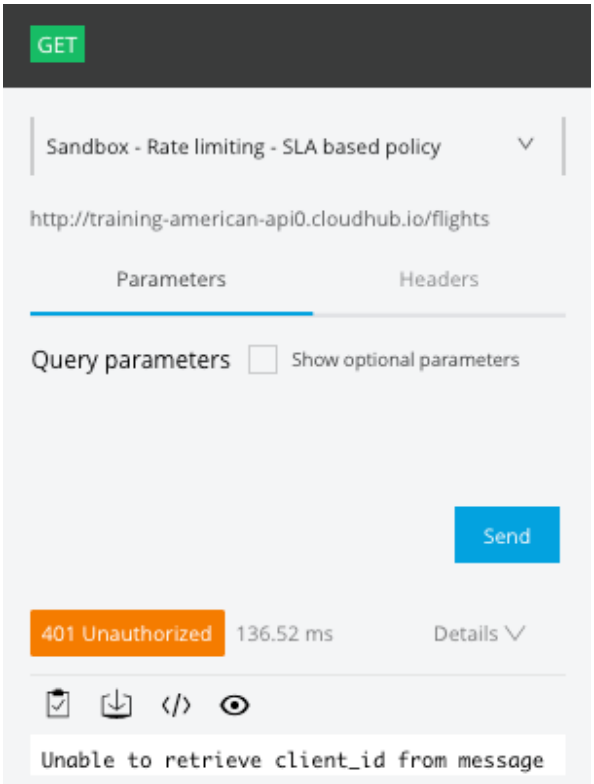
28. Click Apply.
29. In the left-side navigation, click Settings.
30. Change the API instance label to Rate limiting – SLA based policy.



Test the rate limiting – SLA based policy in Exchange

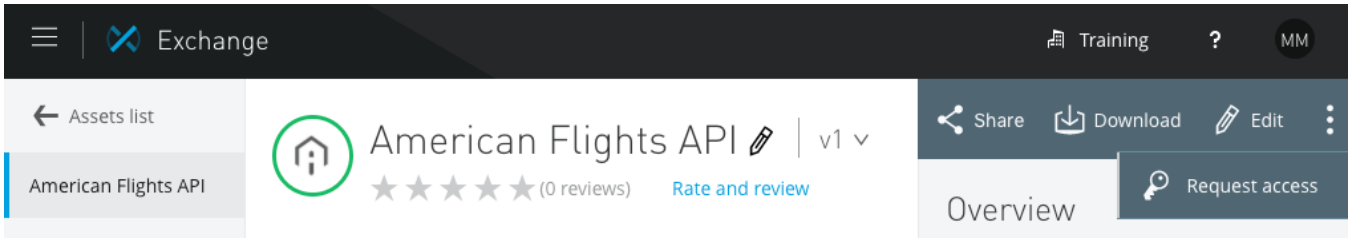
31. Return to the browser tab with your API in Exchange.
32. Refresh the page and select to make a call to the Sandbox – Rate limiting – SLA based policy.

33. Click Send; you should get a 401 response with a message that a client_id could not be retrieved from the message.

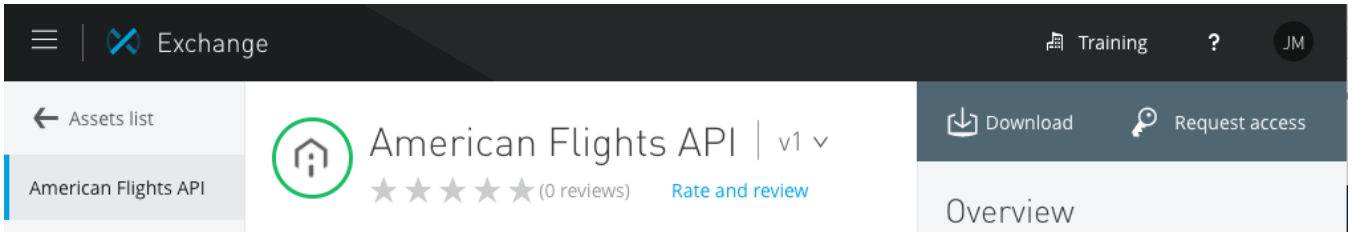


Request access to the API as an internal consumer

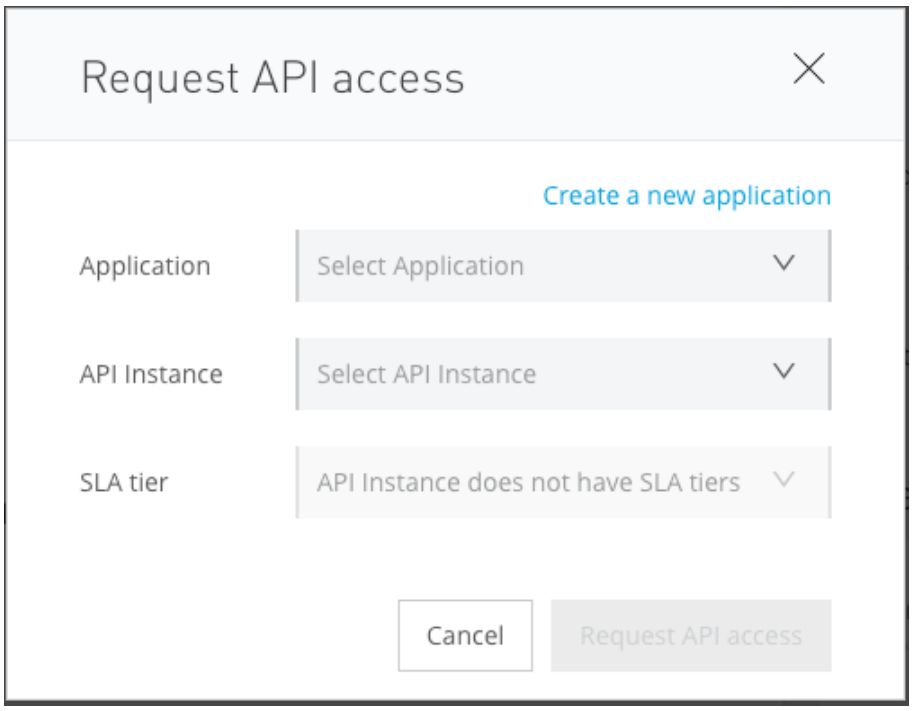
34. In the left-side navigation, click the name of the API to return to its home page.
35. Click the more options button in the upper-right corner and select Request access.



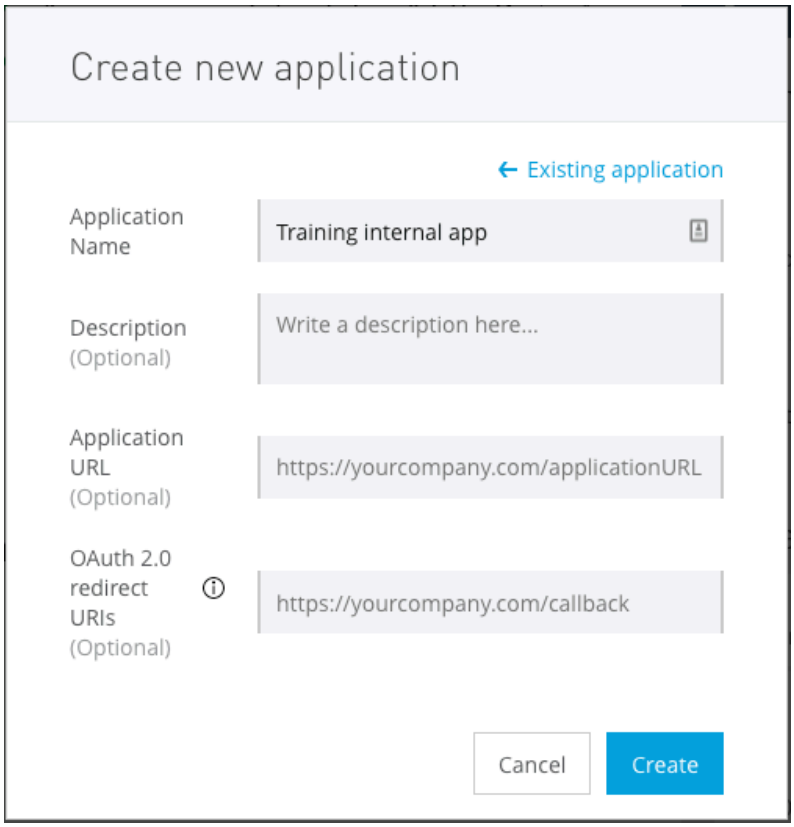
Note: Other internal users that you shared the API with that do not have Edit permissions will see a different menu.



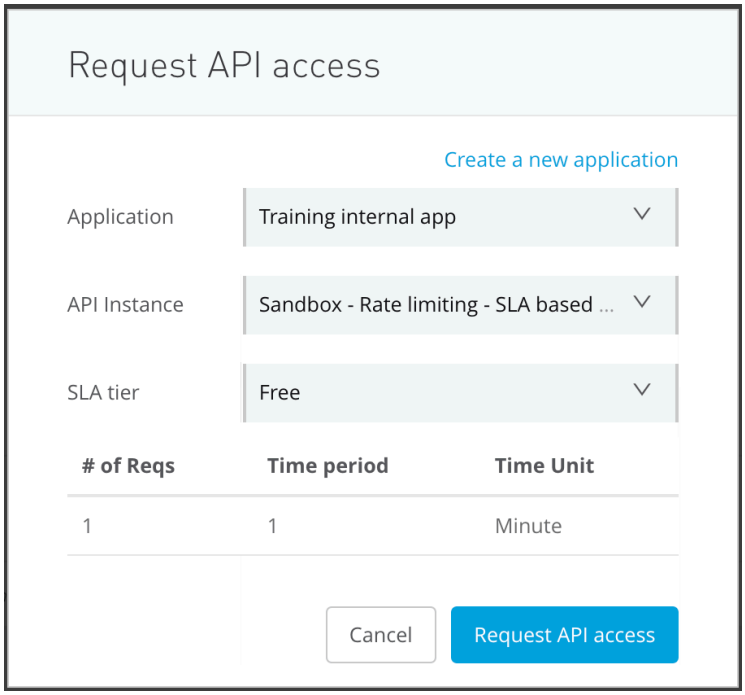
36. In the Request API access dialog box, click the Create a new application link.



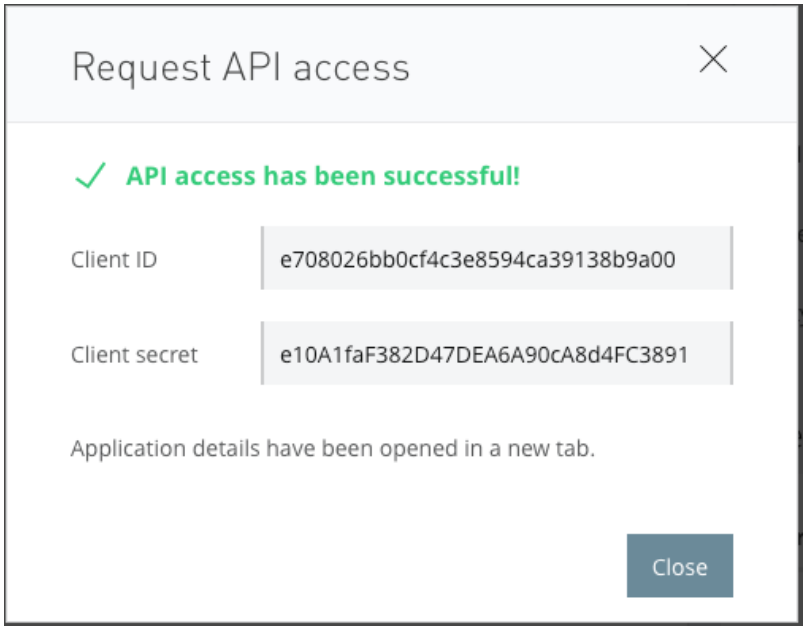
37. In the Create new application dialog box, set the name to Training internal app and click Create.



38. In the Request API access dialog box, set the API instance to Sandbox – Rate limiting SLA - based policy.
39. Set the SLA tier to Free.



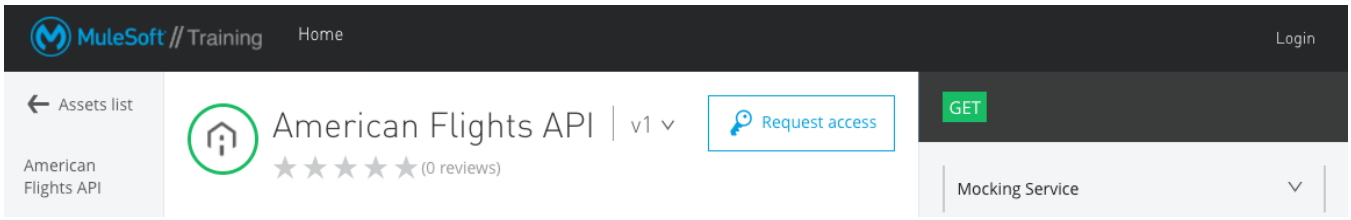
40. Click Request API access.
41. In the Request API access dialog box, view the assigned values for the client ID and client secret.



42. Click Close.

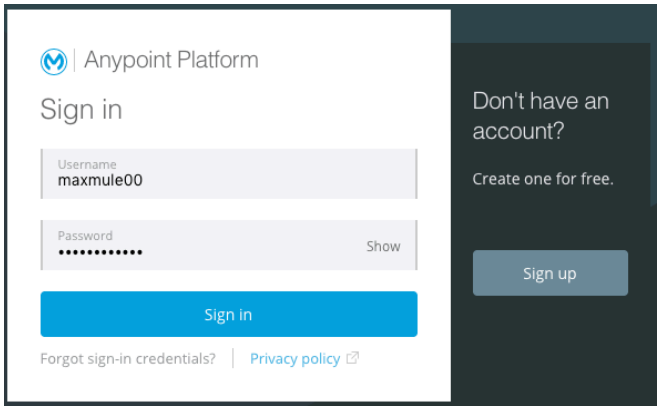
Request access to the API as an external consumer

43. Return to the public portal in the private/incognito window.
44. Refresh the page for the American Flights API; you should now see a Request access button.



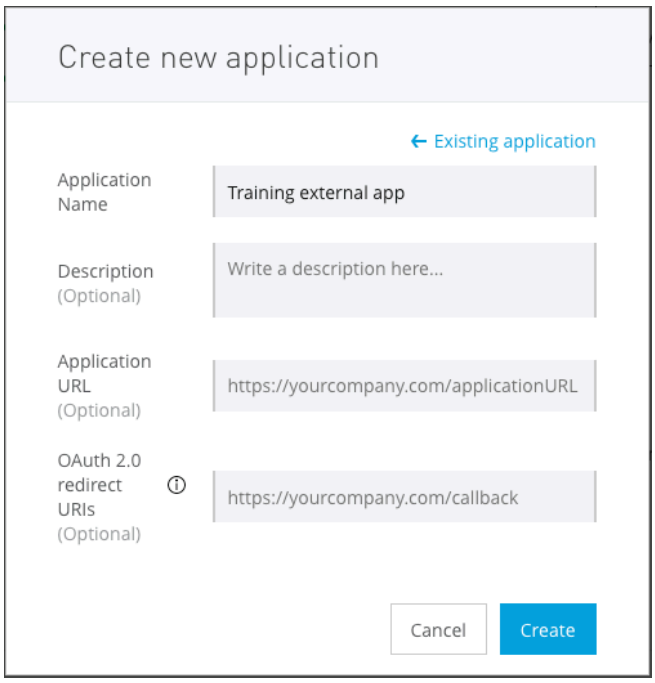
45. Click the Request access button; you should get a page to sign in or create an Anypoint Platform account.
46. Enter your existing credentials and click Sign in.

    Note: Instead of creating an external user, you will just use your existing account.

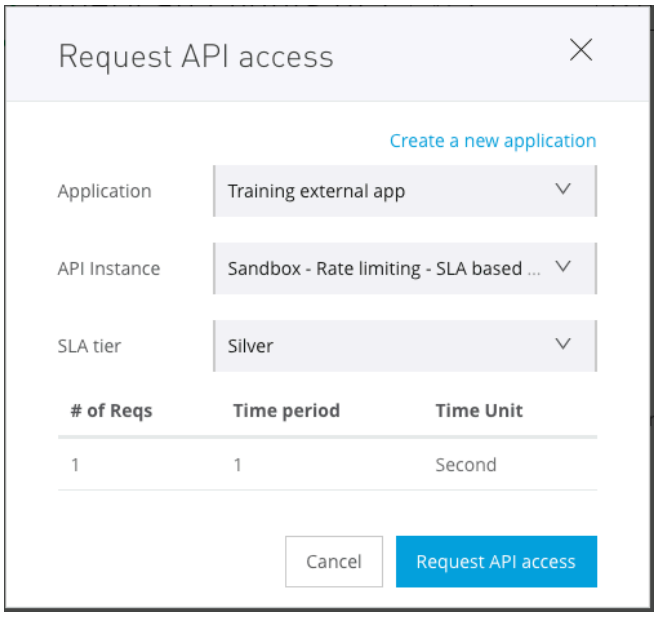


47. Back in the public portal, click the Request access button again.
48. In the Request API access dialog box, click the Create a new application button

49. In the Create new application dialog box, set the name to Training external app and click Create.

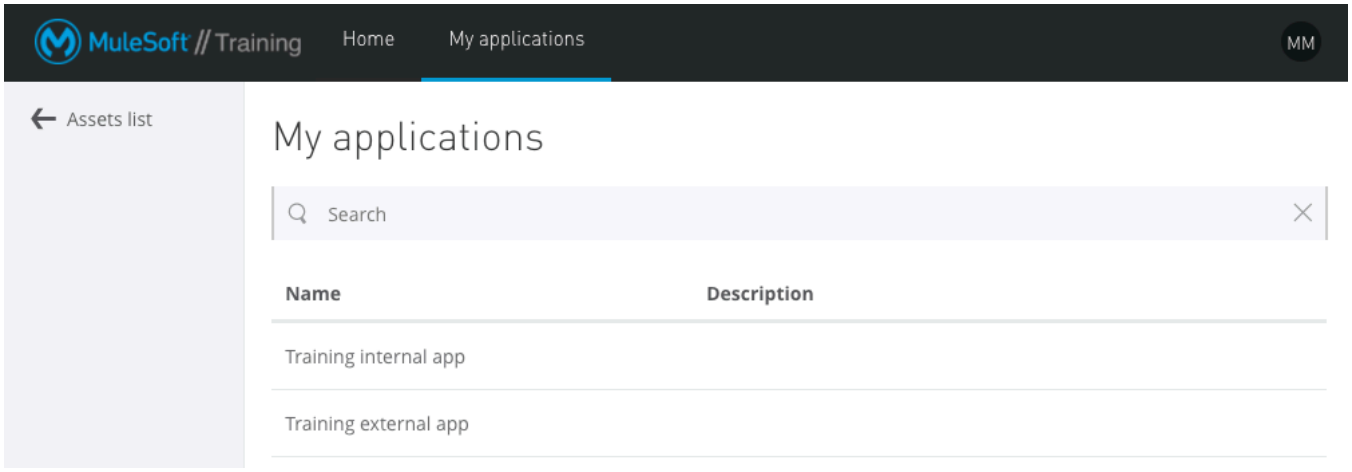


50. Click Create.
51. In the Request API access dialog box, set the API instance to Sandbox – Rate limiting SLA-based policy.
52. Set the SLA tier to Silver.

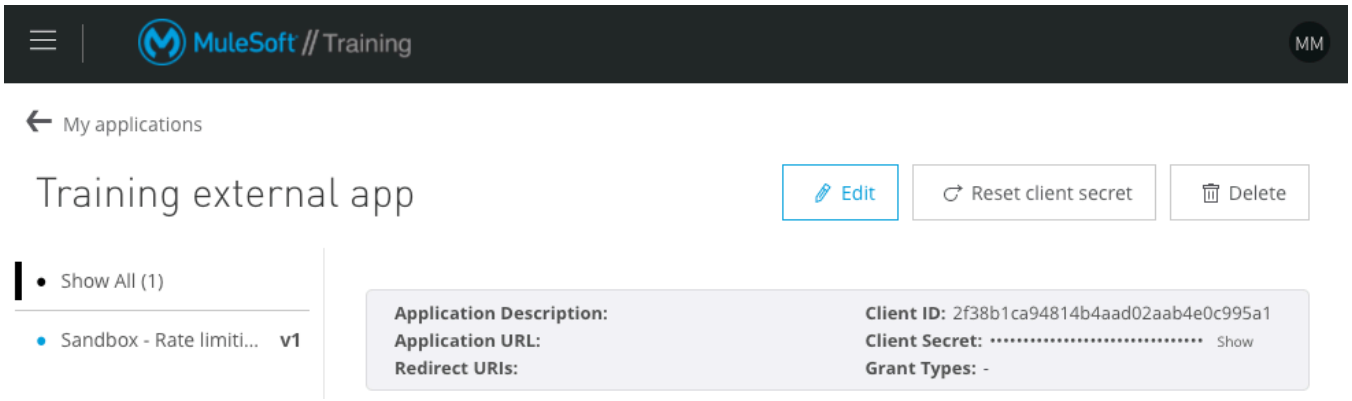


53. Click Request API access.
54. In the Request API access dialog box, click Close.

55. In the portal main menu bar, right-click My applications and select to open it in a new tab; you should see the two applications you created.



56. Click the link for Training external app; you should see what APIs the application has access to, values for the client ID and secret to access them, and request data.

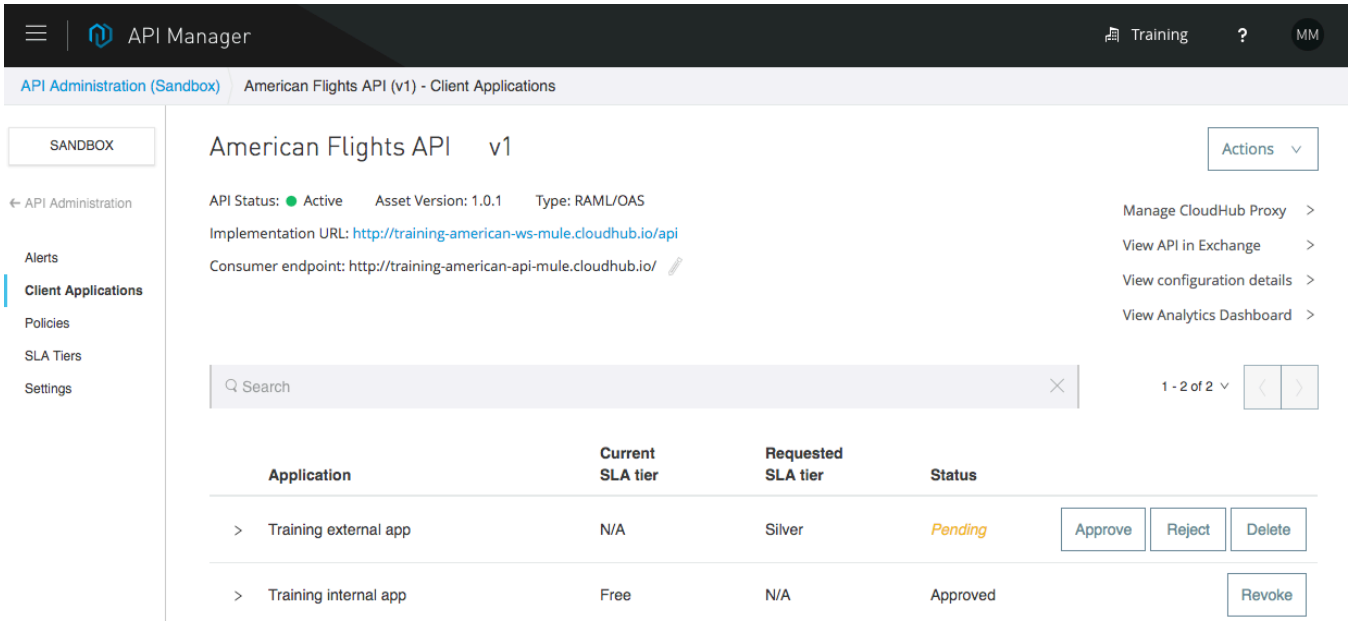


57. Leave this page open in a browser so you can return to it and copy these values.

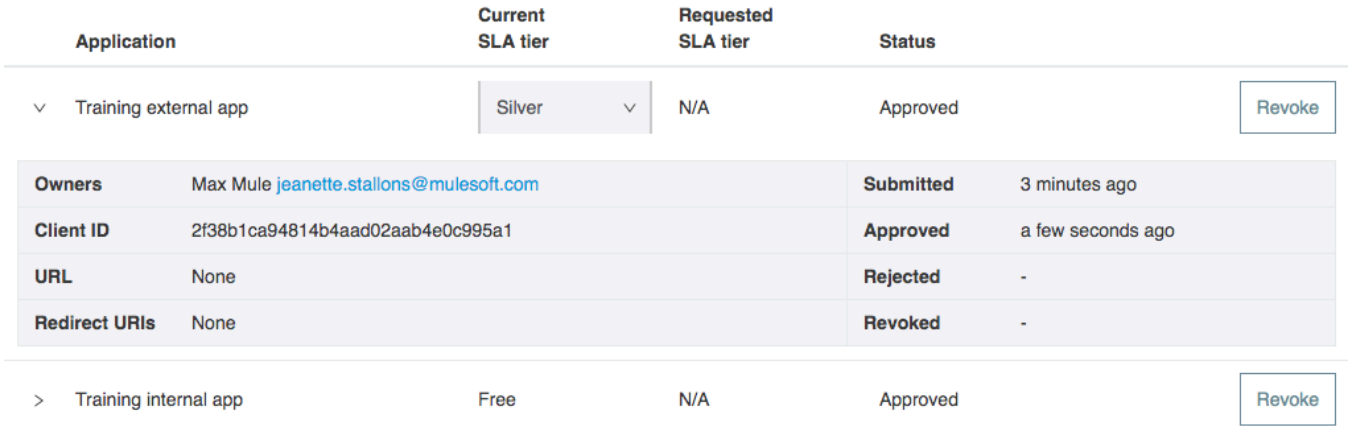
Approve the application requests

58. Return to the browser window and tab with the Settings page for American Flights API (v1) in API Manager.

59. In the left-side navigation, click Client Applications; you should see the two applications that requests access to the API.



60. Click the Approve button for the application requesting Silver tier access.
61. Expand the Training external app row and review its information.

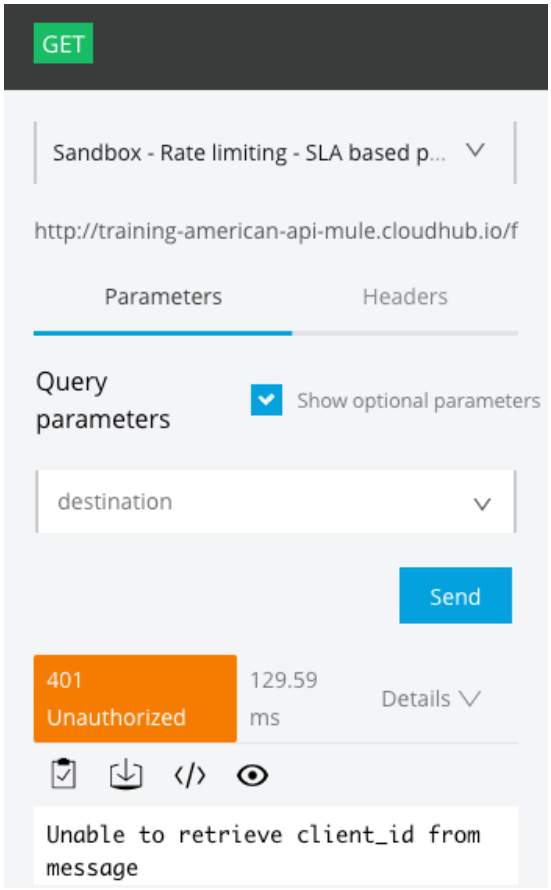


Try to test the rate limiting – SLA based policy from an API portal

62. Return to the browser window and tab with the API console in the public portal.
63. Try again to make a call to the Sandbox – Rate limiting – SLA based policy; you should still get a 401 response with a message that a client_id could not be retrieved from the message.

    Note: There is currently no way to pass the client authentication parameters needed to call this API from an API portal; you will add this in the next walkthrough. You could, however, use

another tool like Postman to make calls to the API, explicitly passing client_id and client_secret query parameters as you did in module 2.

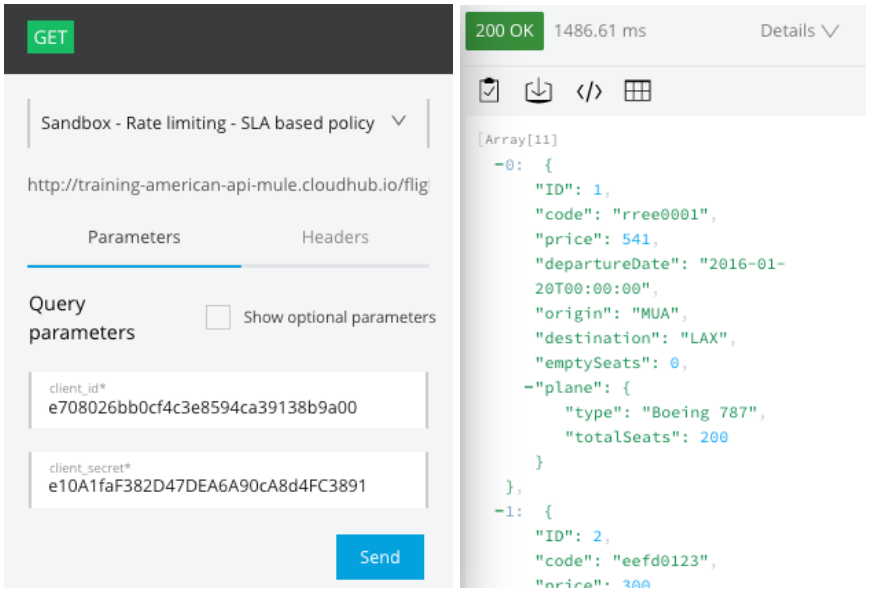


64. Close this browser window.

Walkthrough 5-5: Make calls to an API with a client ID based policy from API portals

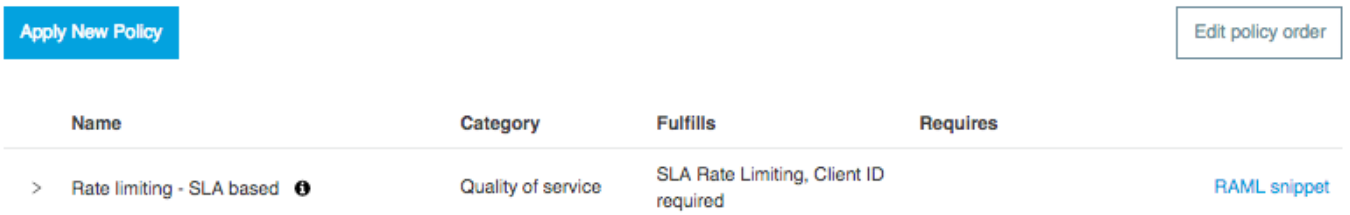
In this walkthrough, you add the ability to make calls to APIs with client ID enforcement policies form internal and public API portals. You will:

- Add authentication query parameters to an API specification.
- Update a managed API to use a new version of an API specification.
- Call a governed API with client credentials from API portals.



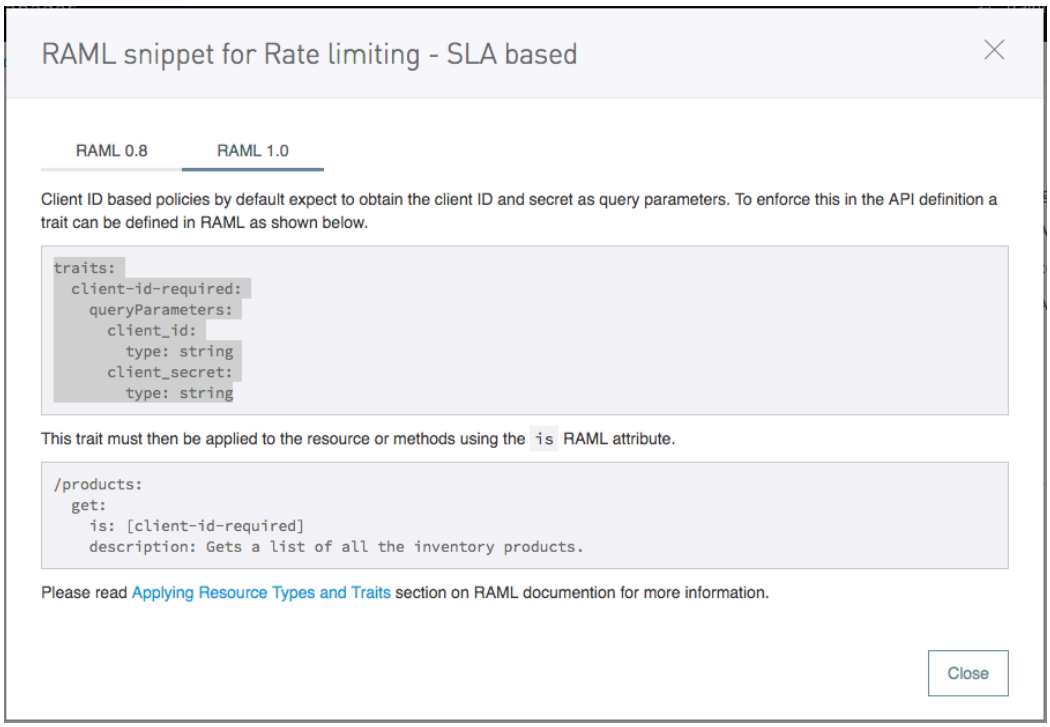
Add authentication query parameters to the API specification

1. Return to the browser tab with the Settings page for American Flights API (v1) in API Manager.
2. In the left-side navigation, click the Policies link.
3. Click the RAML snippet link for the rate limiting – SLA based policy.



4. In the RAML snippet for Rate limiting – SLA based dialog box, select RAML 1.0.

5. Copy the value for the traits.



6. Click Close.
7. Return to the browser tab with your API in Design Center.
8. Go to a new line after the types declaration and paste the traits code you copied.

```
1   #%RAML 1.0
2   version: v1
3   title: American Flights API
4
5   types:
6     AmericanFlight: !include exchange_modu
7
8   traits:
9     client-id-required:
10      queryParameters:
11        client_id:
12          type: string
13        client_secret:
14          type: string
15
16  /flights:
```

9. Go to a new line after the flights resource declaration and add is: [].

```
16  /flights:
17    is: []
18    get:
```

10. Place the cursor inside the array brackets and in the shelf, click client-id-required.

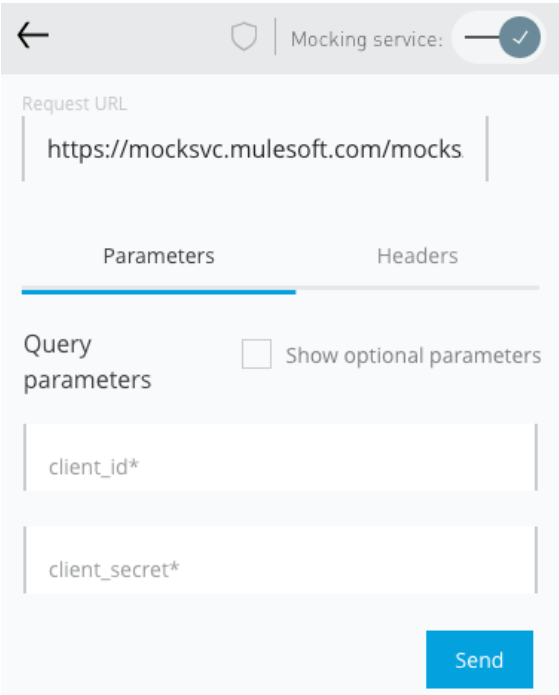
```
16  /flights:
17    is: [client-id-required]
18    get:
```

11. Repeat this process so the trait is applied to all methods of the {ID} resource as well.

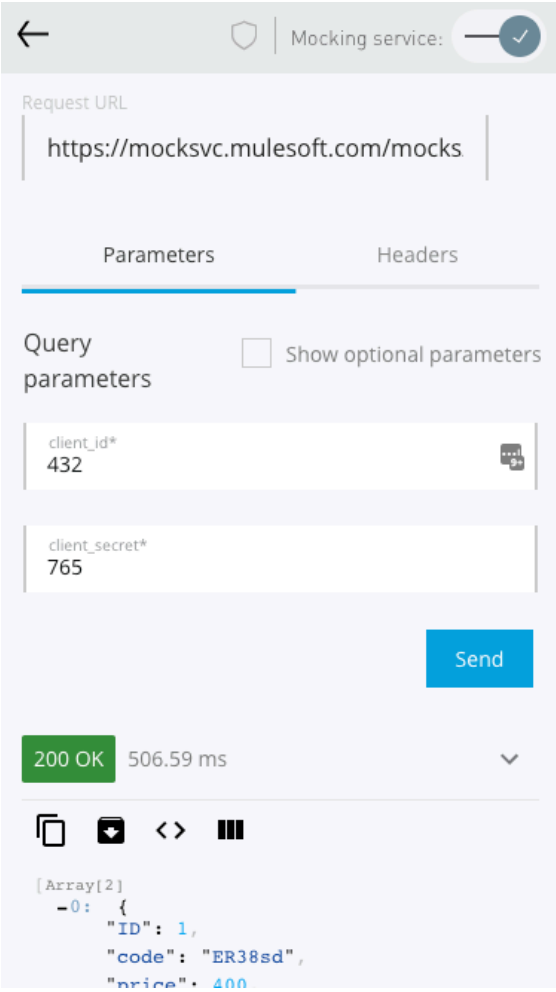
```
45  /{ID}:
46    is: [client-id-required]
47    get:
```

Test the authentication query parameters in API console

12. In the API console, turn on the mocking service.
13. Select to try any one of the resources; you should now see text fields to enter client_id and client_secret query parameters.

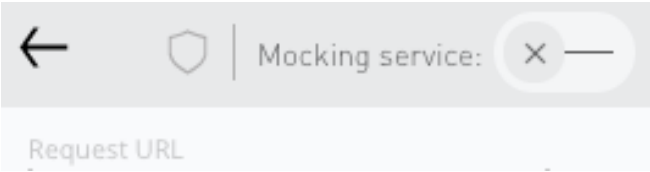


14. Enter any values for the client_id and client_secret and click Send; you should get a 200 response with the example results.



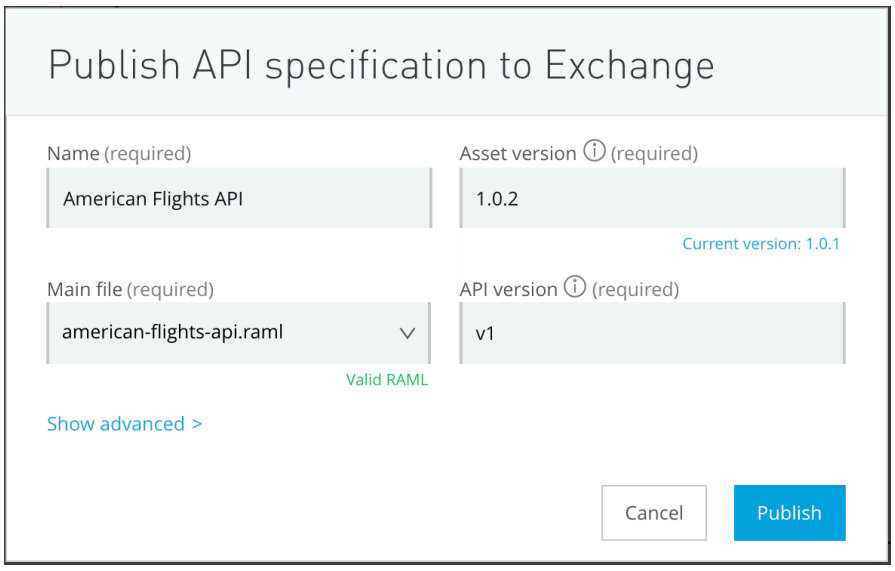
Publish the new version of the API to Exchange

15. Turn off the mocking service.



16. Click the Publish to Exchange button.

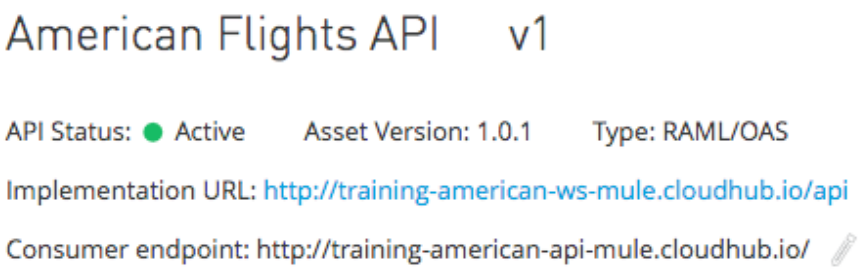
17. In the Publish API specification to Exchange dialog box, note the asset version and click Publish.



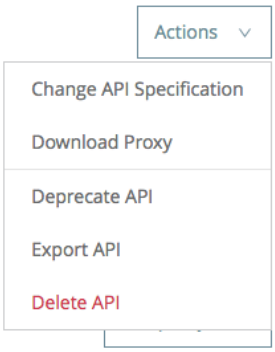
18. After the API is published, click Done in the Publish API specification to Exchange dialog box.

Update the managed API instance to use the new version of the API specification

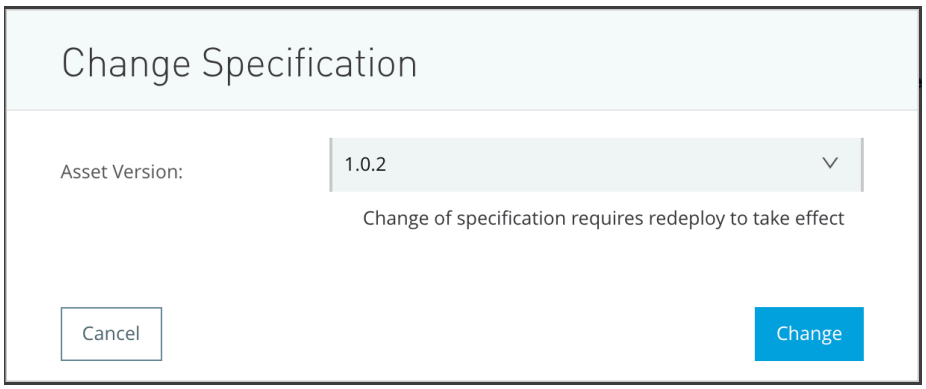
19. Return to browser tab with American Flights API (v1) in API Manager.
20. Locate the asset version displayed at the top of the page; you should see 1.0.1.



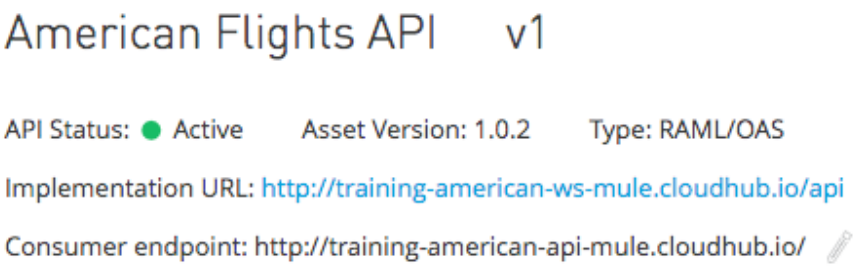
21. Click the Actions button in the upper-right corner and select Change API Specification.



22. In the Change Specification dialog box, select the latest asset version, 1.0.2.
23. Click Change.

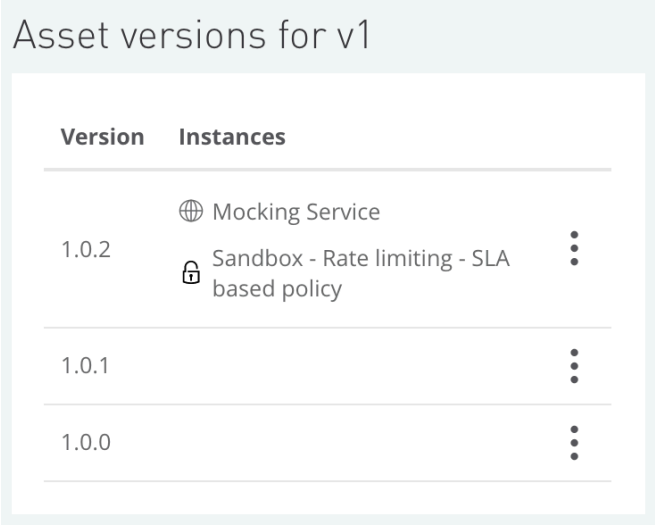


24. Wait until the API Status is green again indicating that the new proxy application has been redeployed.

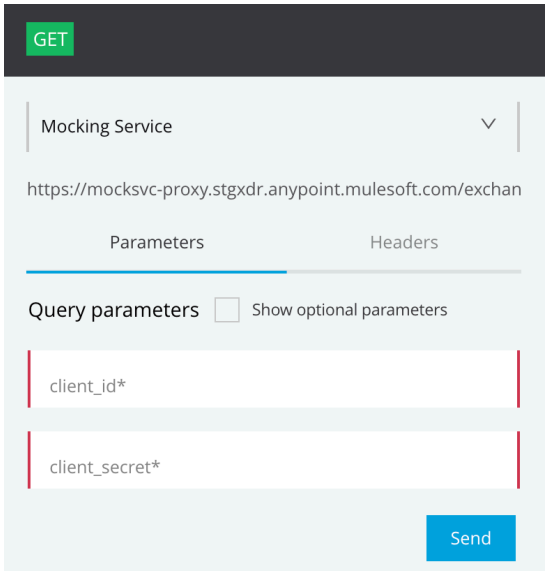


Test the rate limiting – SLA based policy in an API portal with random client authentication values

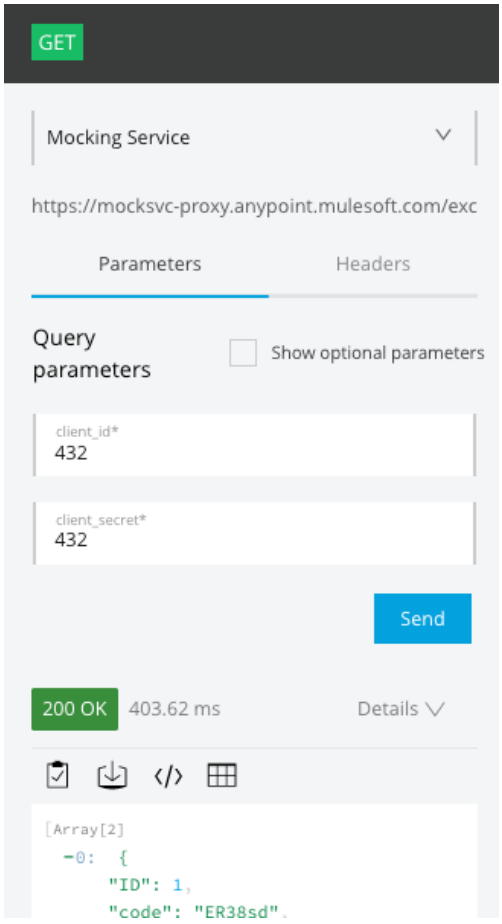
25. Return to the browser tab with Exchange.
26. Return to the home page for the API; you should see the new asset version listed.



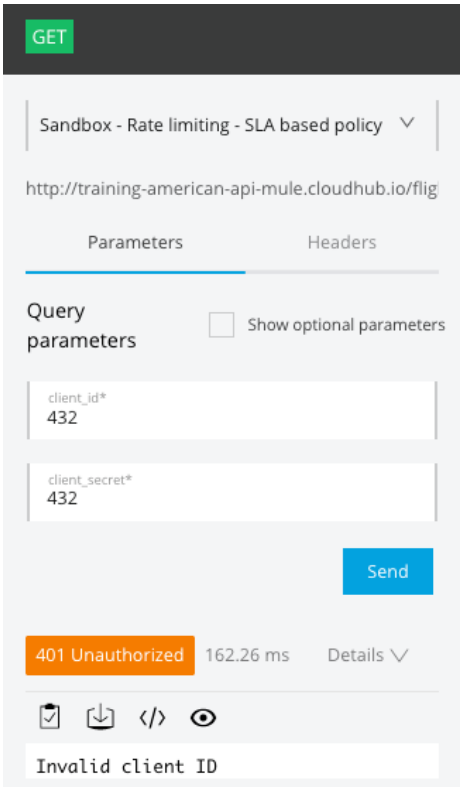
27. Click the GET method for the flights resource; you should now see required text boxes to enter the required client_id and client_secret values.



28. With the Mocking Service endpoint selected, enter any values for the client_id and client_secret.
29. Click Send; you should get results.

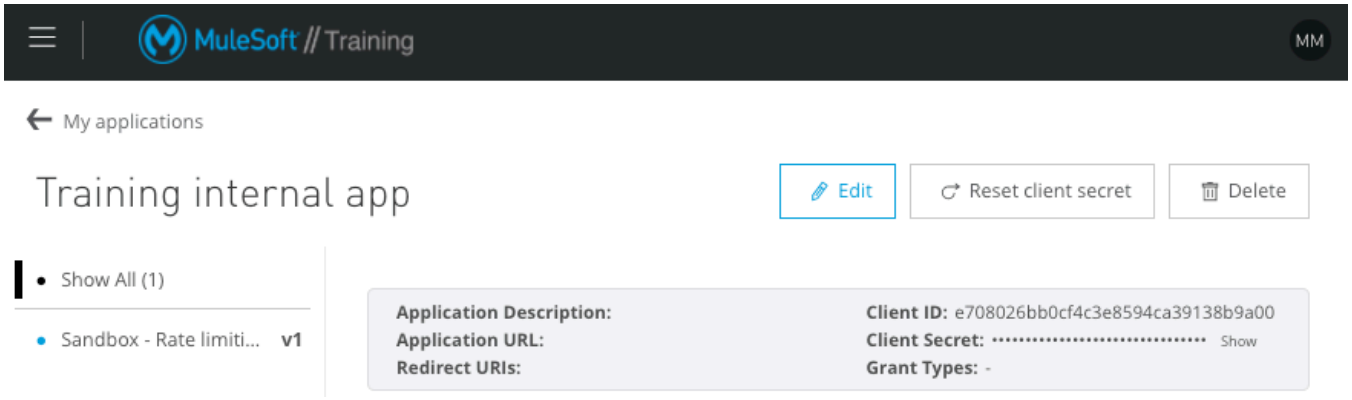


30. Change the API instance to Sandbox – Rate limiting – SLA based policy and leave the random values for the authentication parameters.
31. Click Send; you should now get a 410 response with a message that the client ID is invalid.

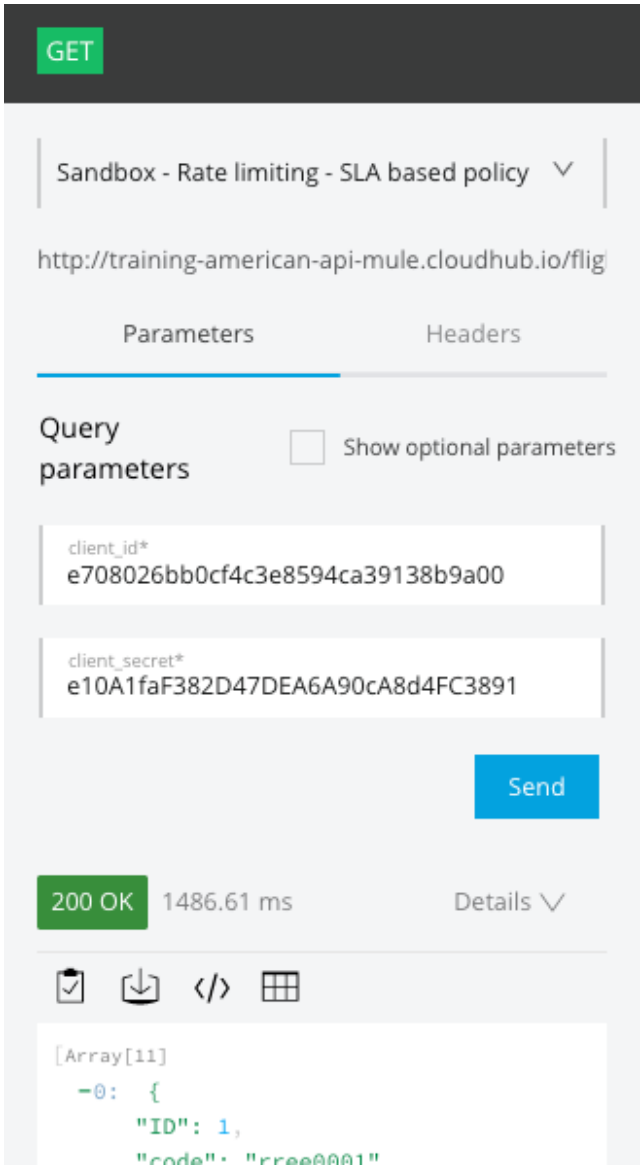


Test the rate limiting – SLA based policy in an API portal with assigned client authentication values

32. In the left-side navigation, click Assets list.
33. In the left-side navigation, click Public Portal.
34. In the main menu, select My applications.
35. Click the Training internal application.



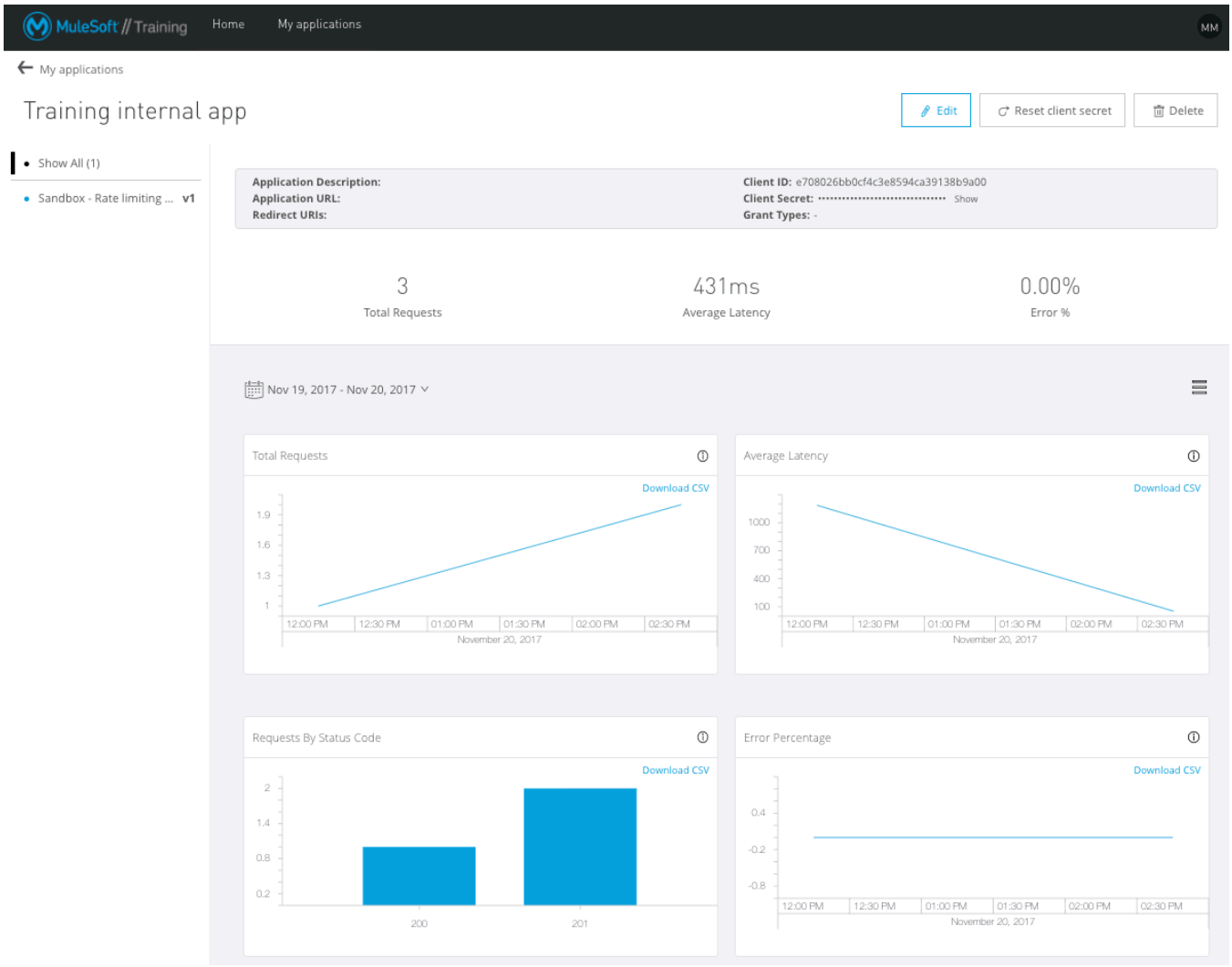
36. Copy the value of its client ID.
37. Right-click Home in the main menu bar and select to open the link in a new browser tab.
38. In that new tab, click the American Flights API.
39. Click the GET method for the flights resource.
40. In the API console, change the API instance to Sandbox – rate limiting – SLA based policy.
41. In the client_id field, paste the value of the client ID.
42. Return to the other browser tab and copy the value of the client secret.
43. Return to the other browser tab and paste the value of the client secret.
44. Click Send; you should now get a 200 response with live results.



45. Make multiple calls to this method.

Review request data

46. Return to the other browser tab with the application information.
47. Change the date selection to Last Day; you should see data about your requests.



48. Close all the browser tabs with Anypoint Platform.