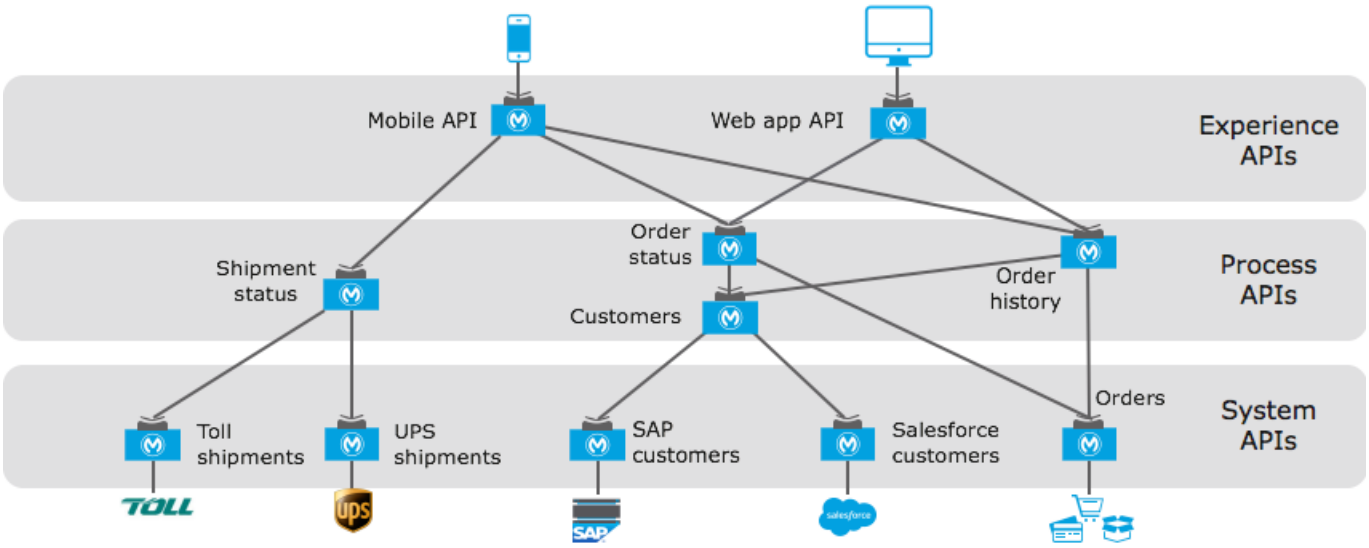
Module 1: Introducing Application Networks and API-Led Connectivity



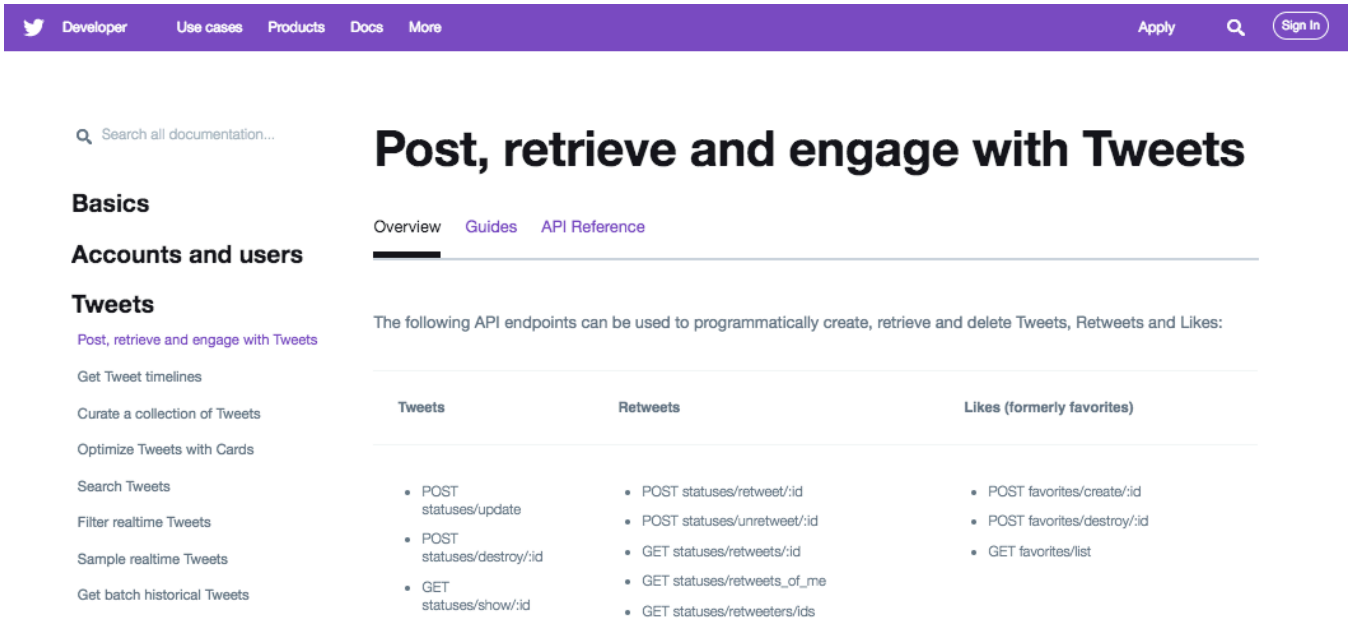
At the end of this module, you should be able to:

- Explain what an application network is and its benefits.
- Describe how to build an application network using API-led connectivity.
- Explain what web services and APIs are.
- Explore API directories and references.
- Make calls to secure and unsecured APIs.

Walkthrough 1-1: Explore an API directory and an API reference

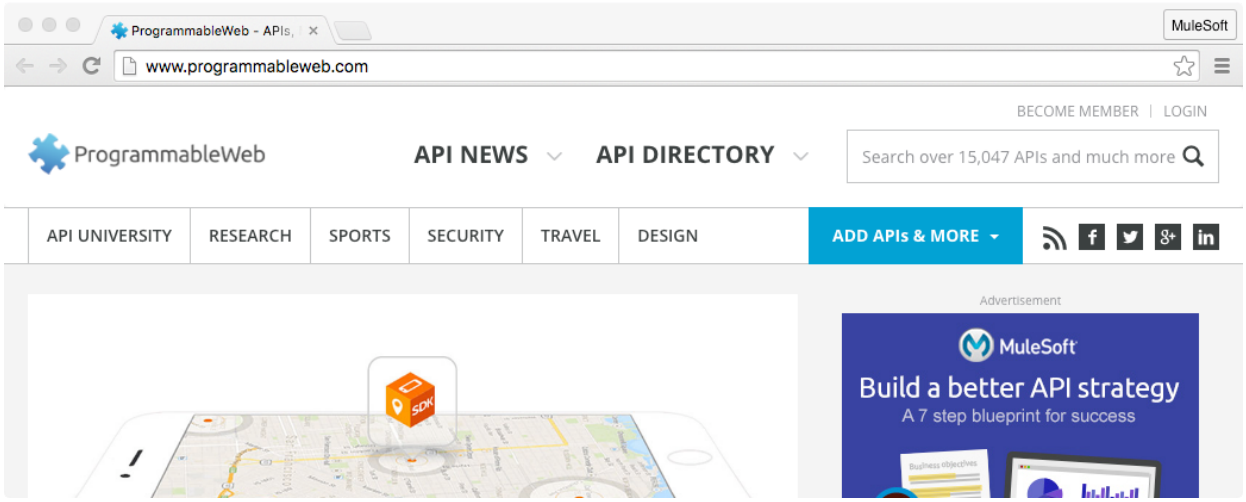
In this walkthrough, you make calls to a RESTful API. You will:

- Browse the ProgrammableWeb API directory.
- Explore the API Reference for the Twitter API.

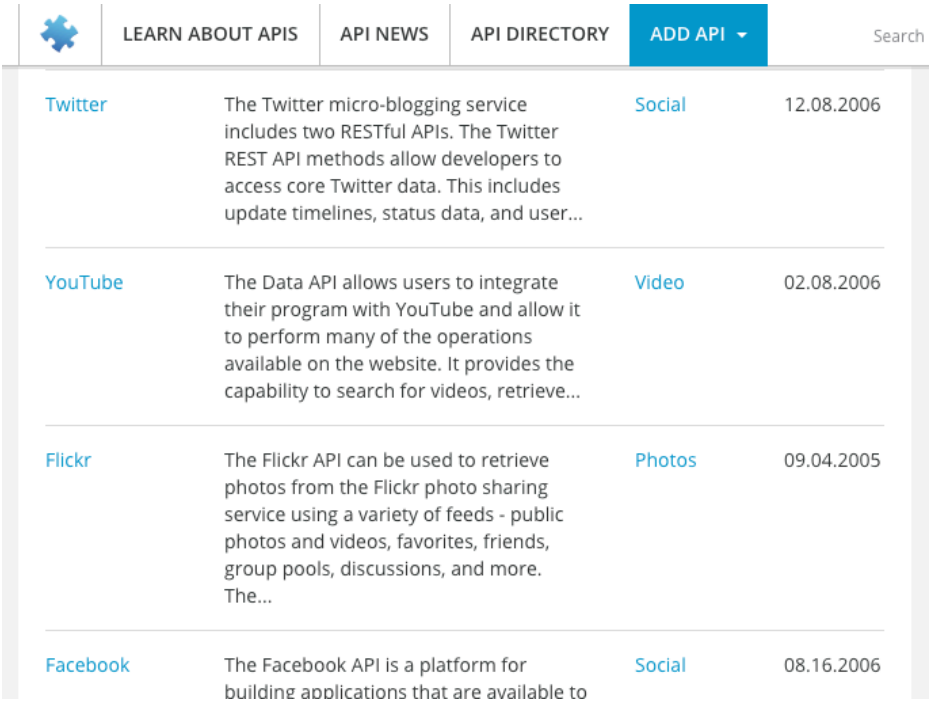


Browse the ProgrammableWeb API directory

1. In a web browser, navigate to http://www.programmableweb.com/.
2. Click the API directory link.



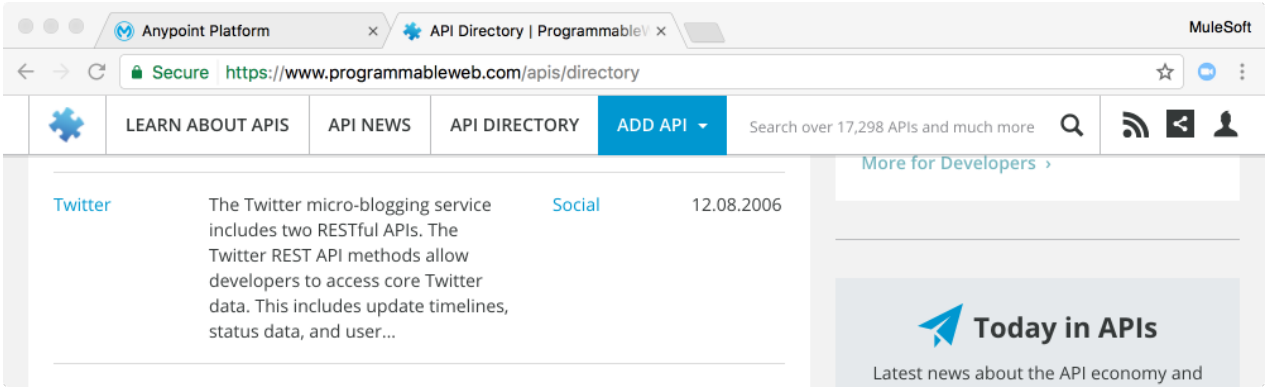
3. Browse the list of popular APIs.



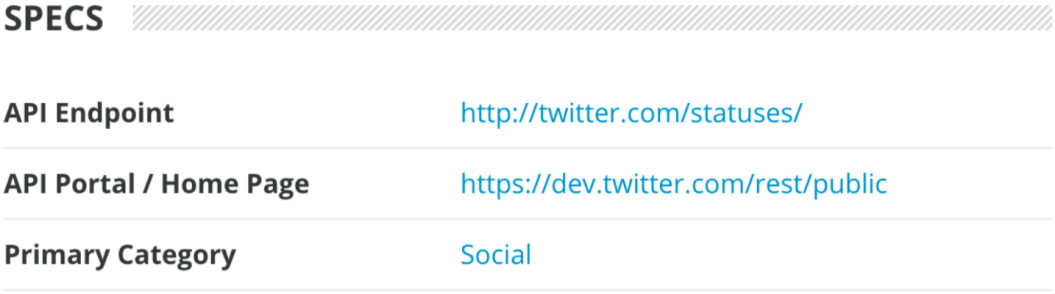
Explore the API reference for the Twitter API

4. Scroll down and click the link for the Twitter API.

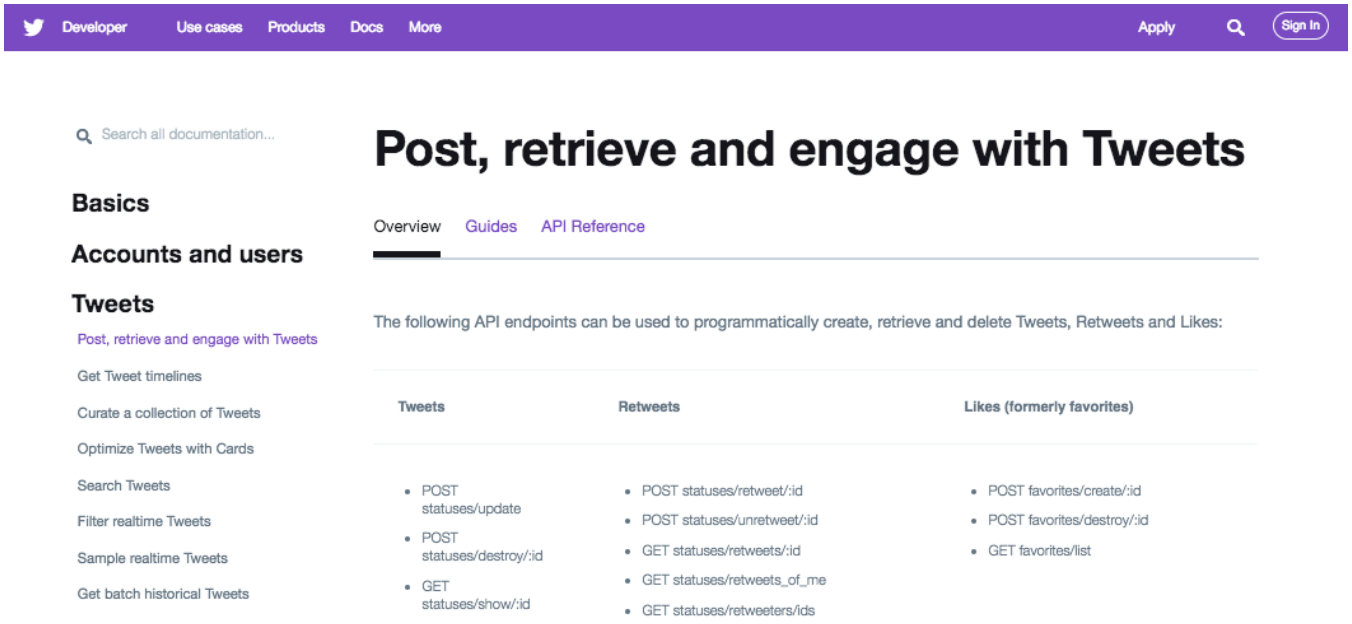
   Note: If Twitter is no longer displayed on the main page, search for it.



5. In the Specs section, click the API Portal / Home Page link.

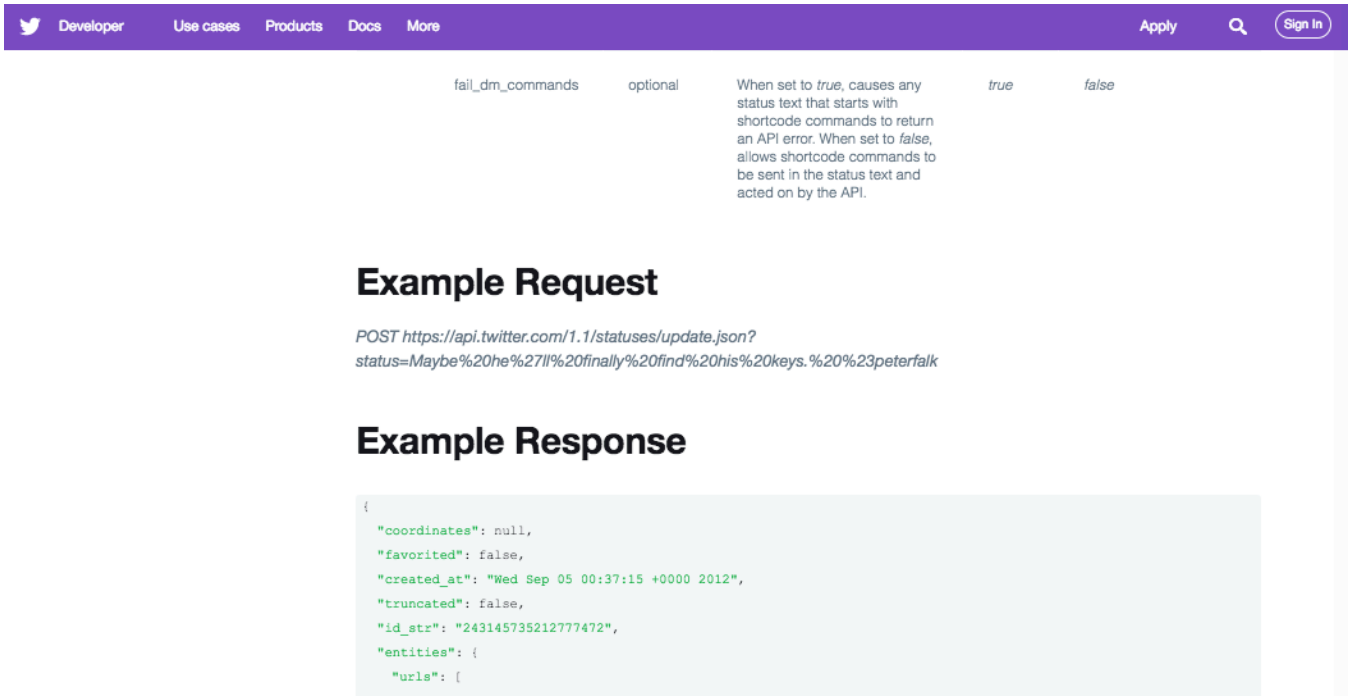


6. In the new browser tab that opens, click Tweets in the left-side navigation.
7. In the left-side navigation, click Post, retrieve and engage with Tweets.



8. Browse the list of requests you can make to the API.
9. Select the API Reference tab beneath the title of the page.

10. Review the information for POST statuses/update including parameters, example request, and example response.

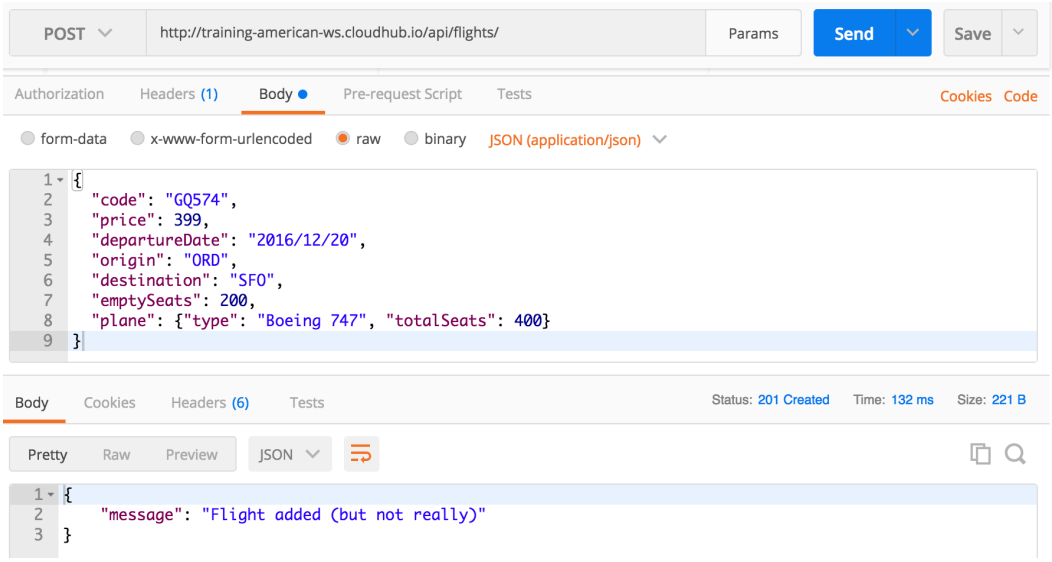


11. Close the two browser tabs.

Walkthrough 1-2: Make calls to an API

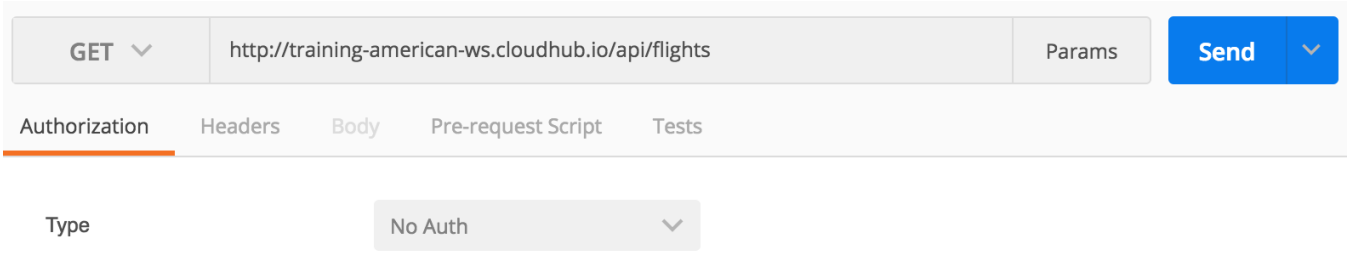
In this walkthrough, you make calls to a RESTful API. You will:

- Use Postman to make calls to an unsecured API (an implementation).
- Make GET, DELETE, POST, and PUT calls.
- Use Postman to make calls to a secured API (an API proxy).

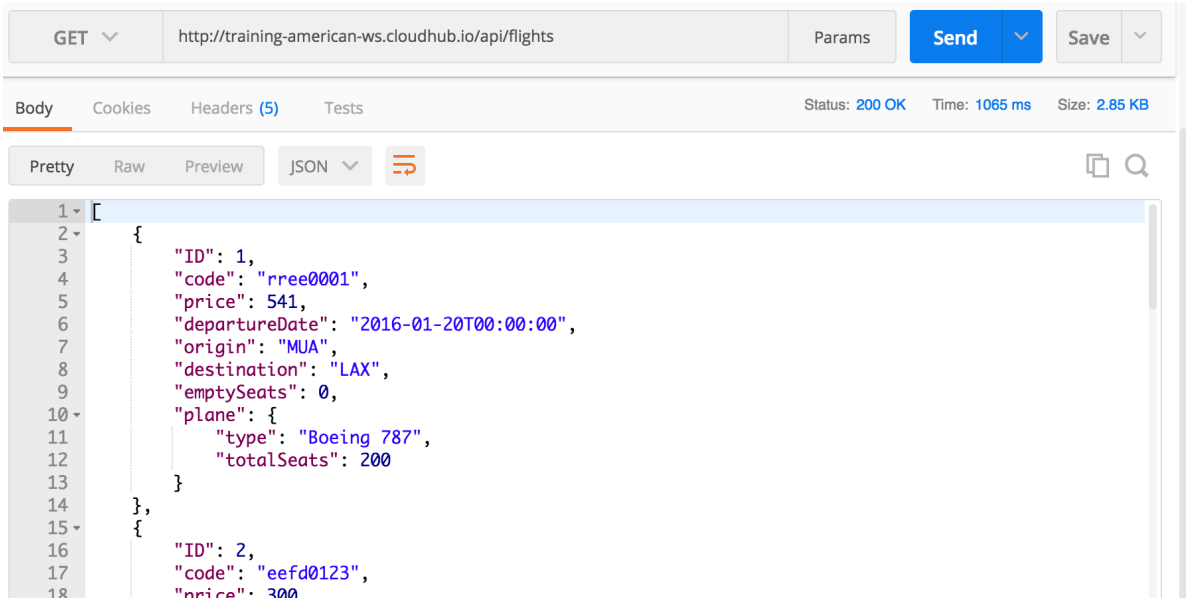


Make GET requests to retrieve data

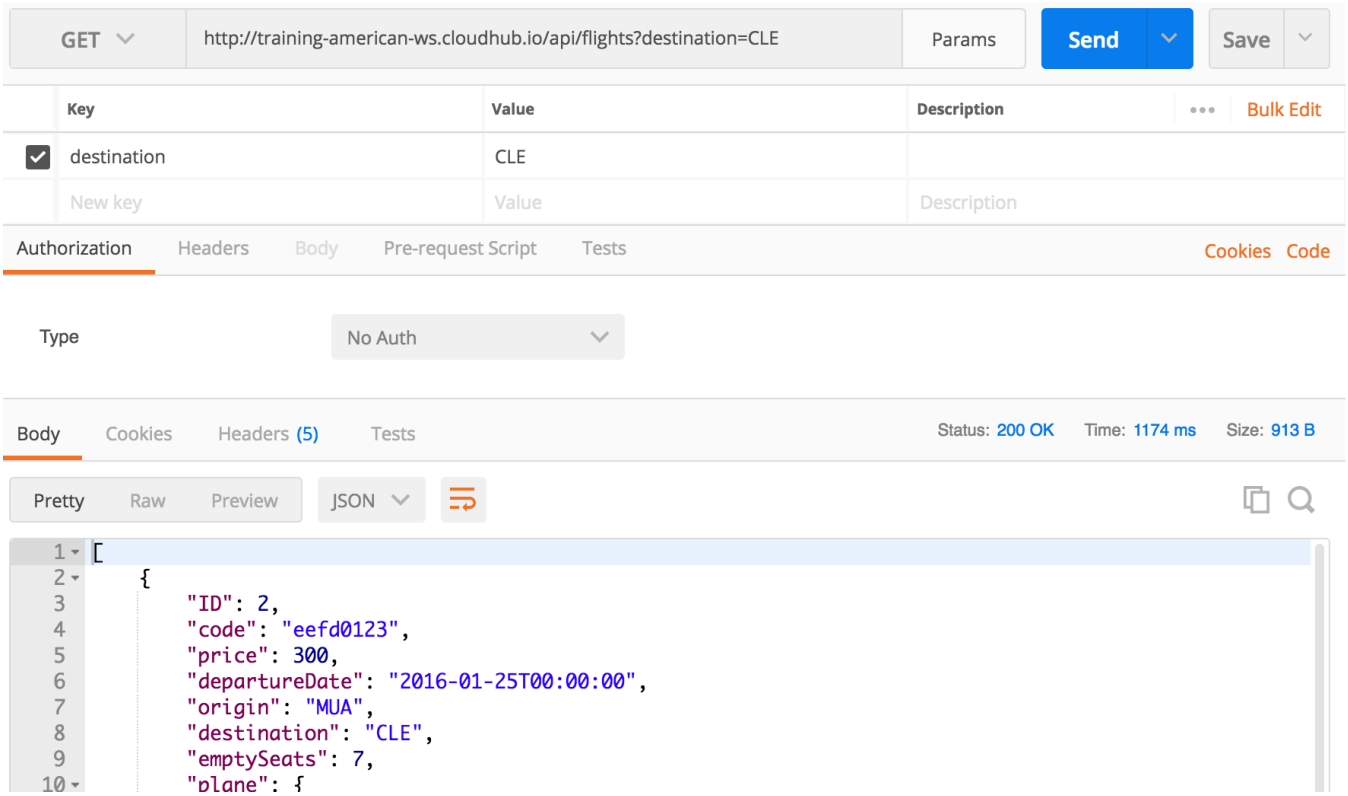
1. Return to or open Postman.
2. Make sure the method is set to GET.
3. Return to the course snippets.txt file.
4. Copy the URL for the American Flights web service: http://training-american-ws.cloudhub.io/api/flights.
5. Return to Postman and paste the URL in the text box that says Enter request URL.



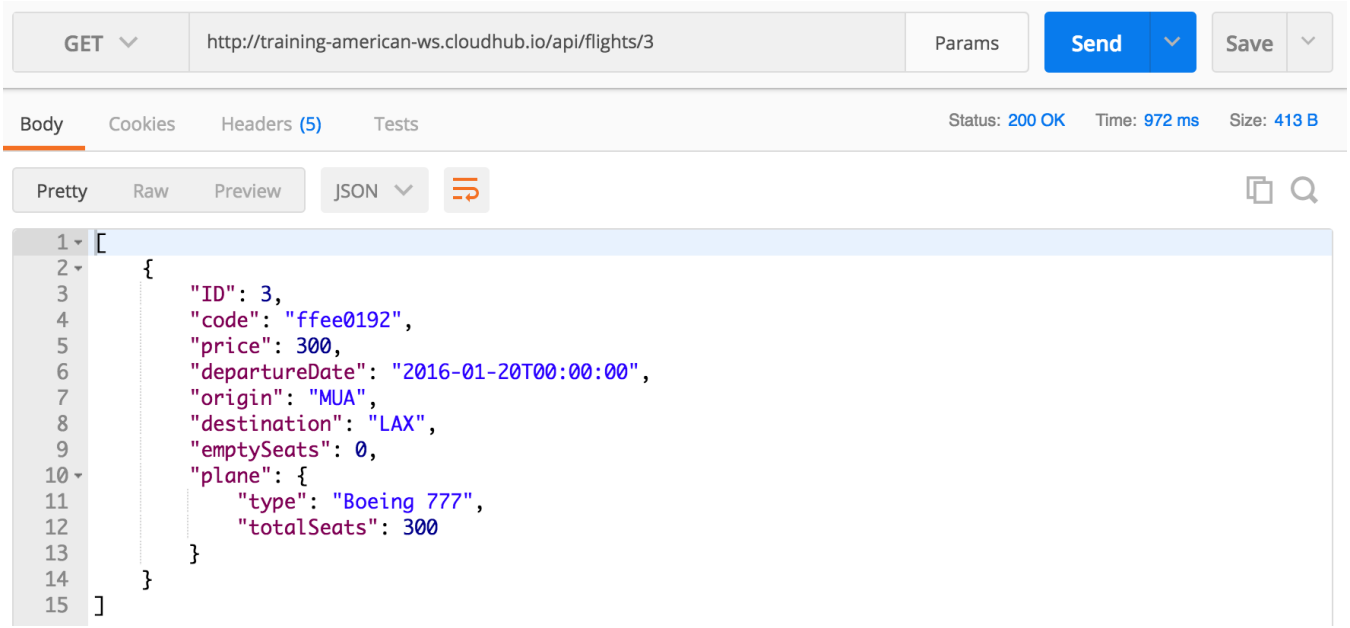
6. Click the Send button; you should get a response.
7. Locate and click the return HTTP status code of 200.
8. Review the response body containing flights to SFO, LAX, and CLE.



9. Click the Params button next to the URL.
10. In the area that appears, set the key to destination and the value to CLE.
11. Click the Send button; you should get just flights to CLE returned.



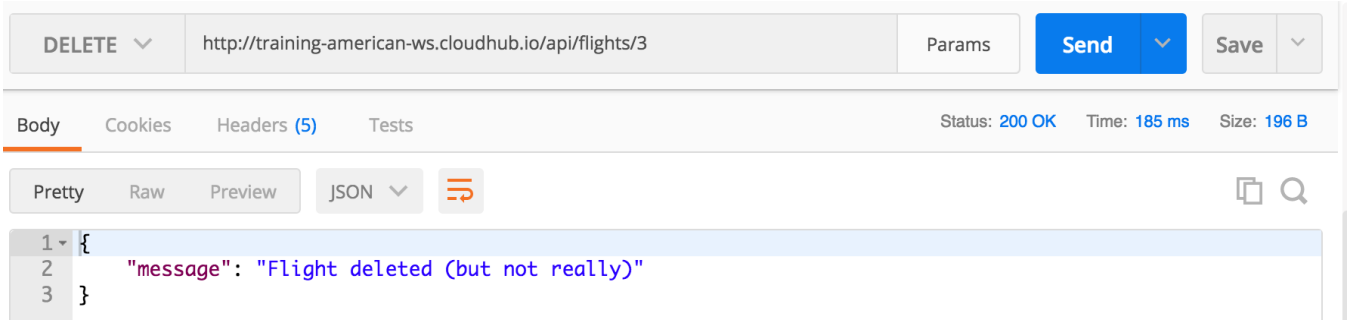
12. Click the X next to the parameter to delete it.
13. Change the request URL to use a uri parameter to retrieve the flight with an ID of 3: http://training-american-ws.cloudhub.io/api/flights/3
14. Click the Send button; you should see only the flight with that ID returned.



Make DELETE requests to delete data

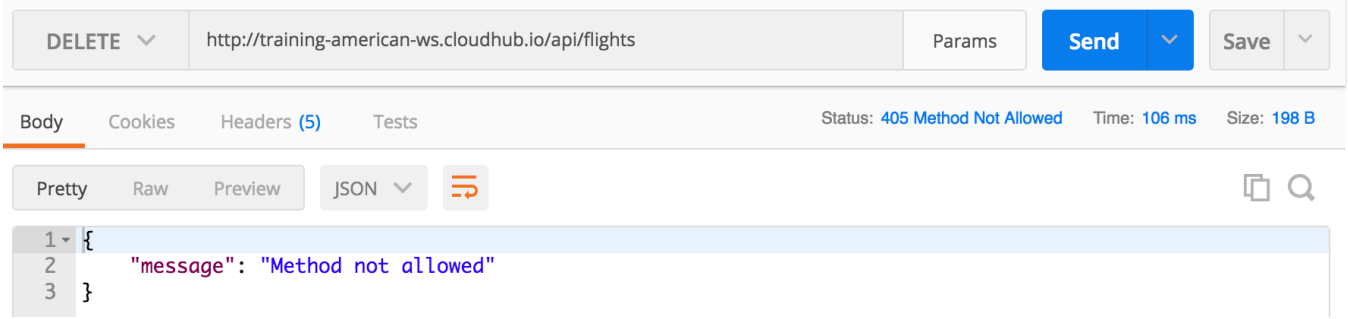
15. Change the method to DELETE.
16. Click the Send button; you should see a 200 response with a message that the Flight was deleted (but not really).

    Note: The database is not actually modified so that its data integrity can be retained for class.



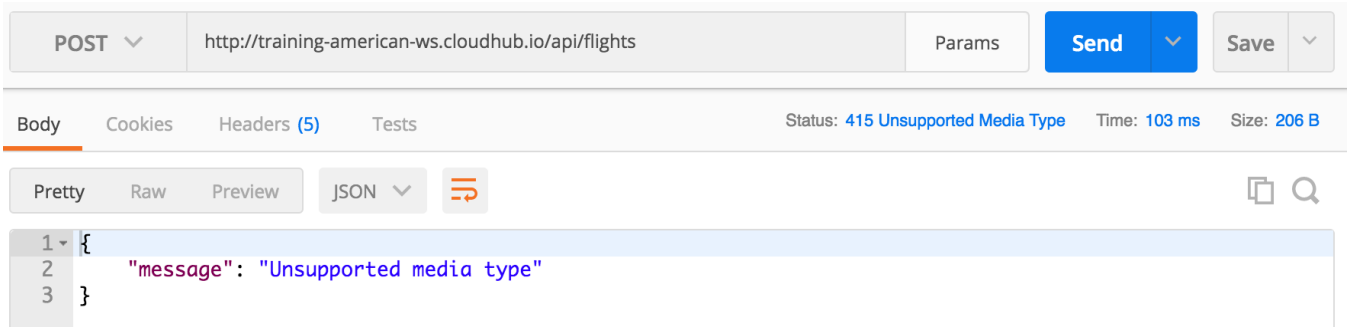
17. Remove the URI parameter from the request: http://training-american-ws.cloudhub.io/api/flights.

18. Click the Send button; you should get a 405 response with a message of method not allowed.

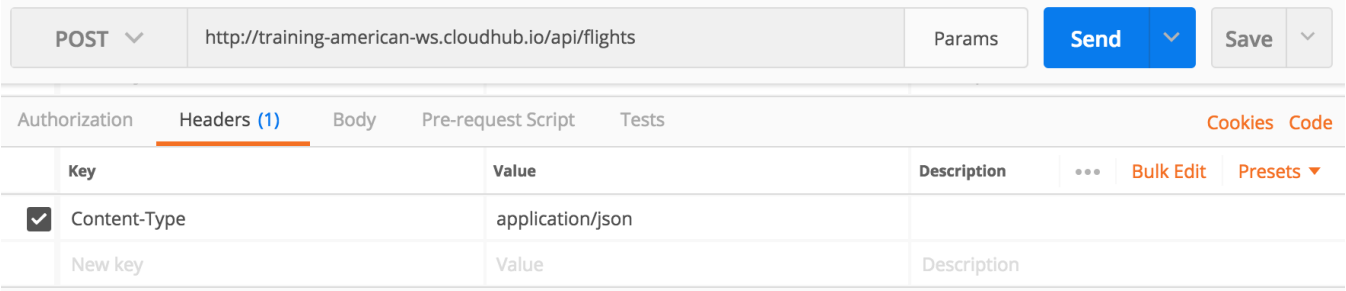


Make a POST request to add data

19. Change the method to POST.
20. Click the Send button; you should get a 415 response with a message of unsupported media type.

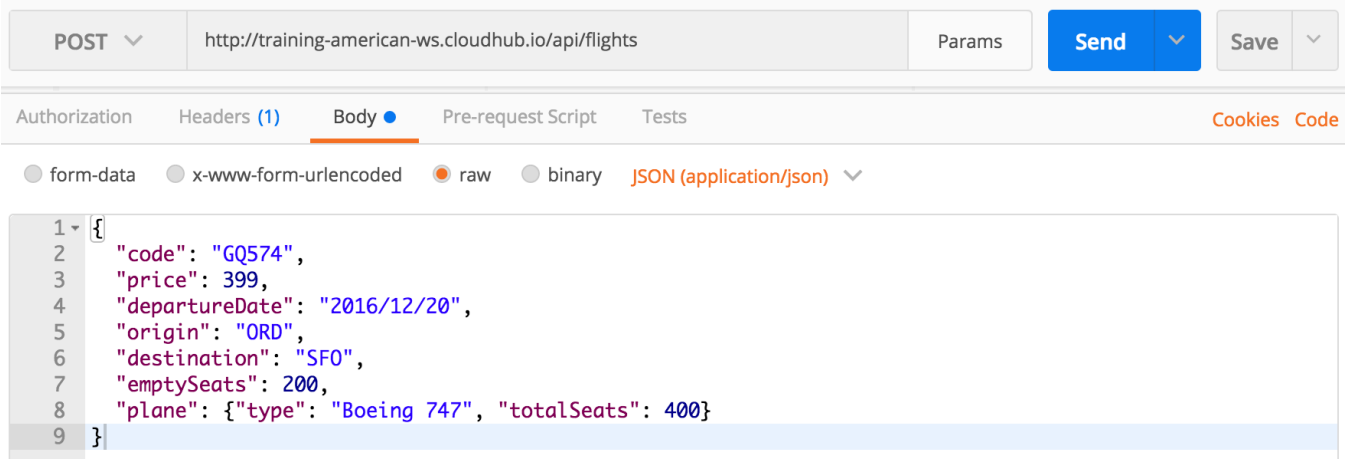


21. Click the Headers link under the request URL.
22. Click in the Headers key field, type C, and then select Content-Type.
23. Click in the Value field, type A, and then select application/json.

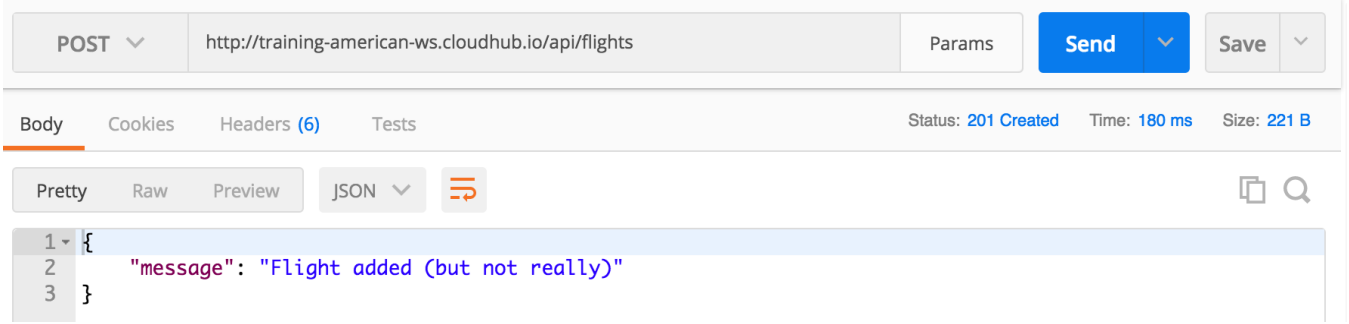


24. Click the Body link under the request URL.
25. Select the raw radio button.
26. Return to the course snippets.txt file and copy the value for American Flights API post body.

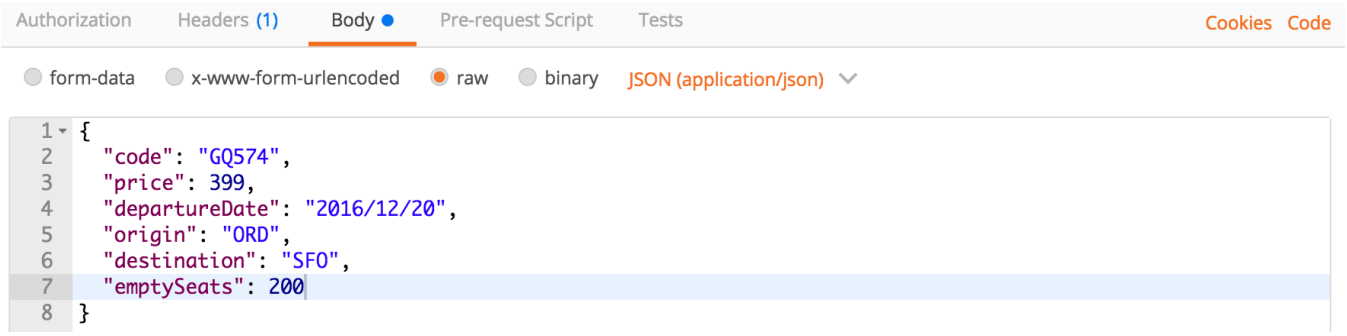
27. Return to Postman and paste the code in the body text area.



28. Click the Send button; you should see a 201 response with the message Flight added (but not really).

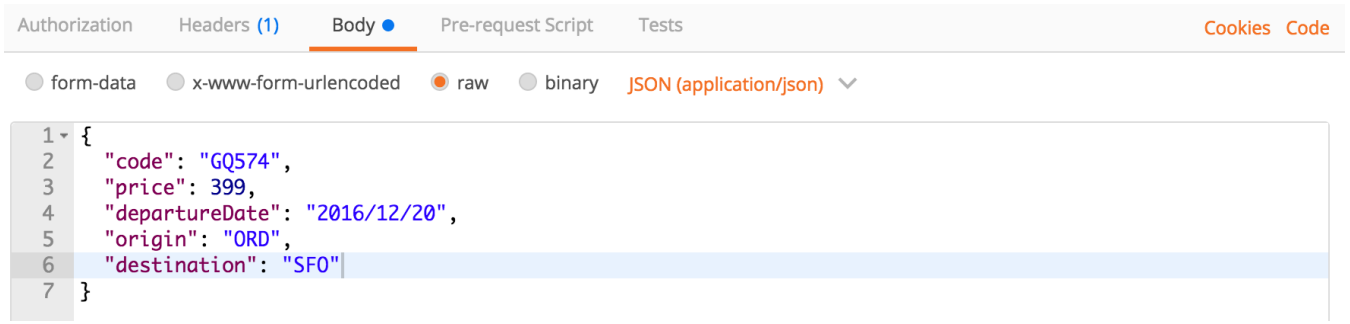


29. Return to the request body and remove the plane field and value from the request body.
30. Remove the comma after the emptySeats key/value pair.

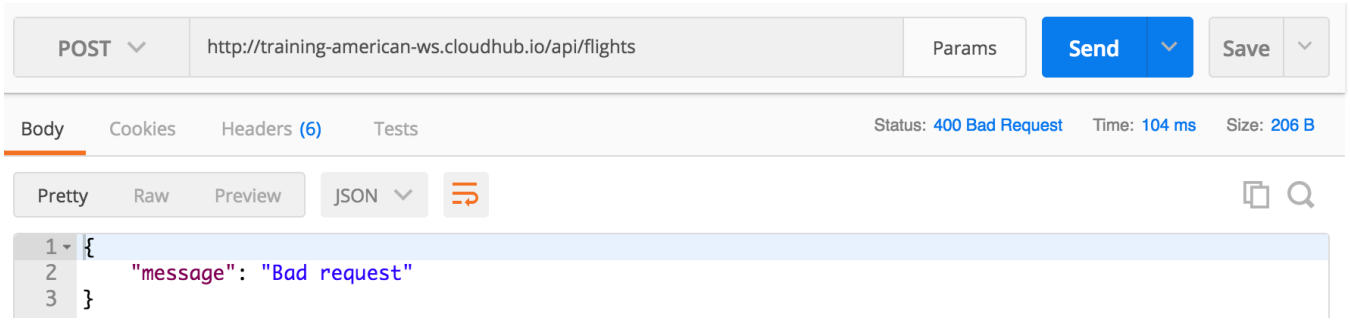


31. Send the request; the message should still post successfully.

32. In the request body, remove the emptySeats key/value pair.
33. Delete the comma after the destination key/value pair.



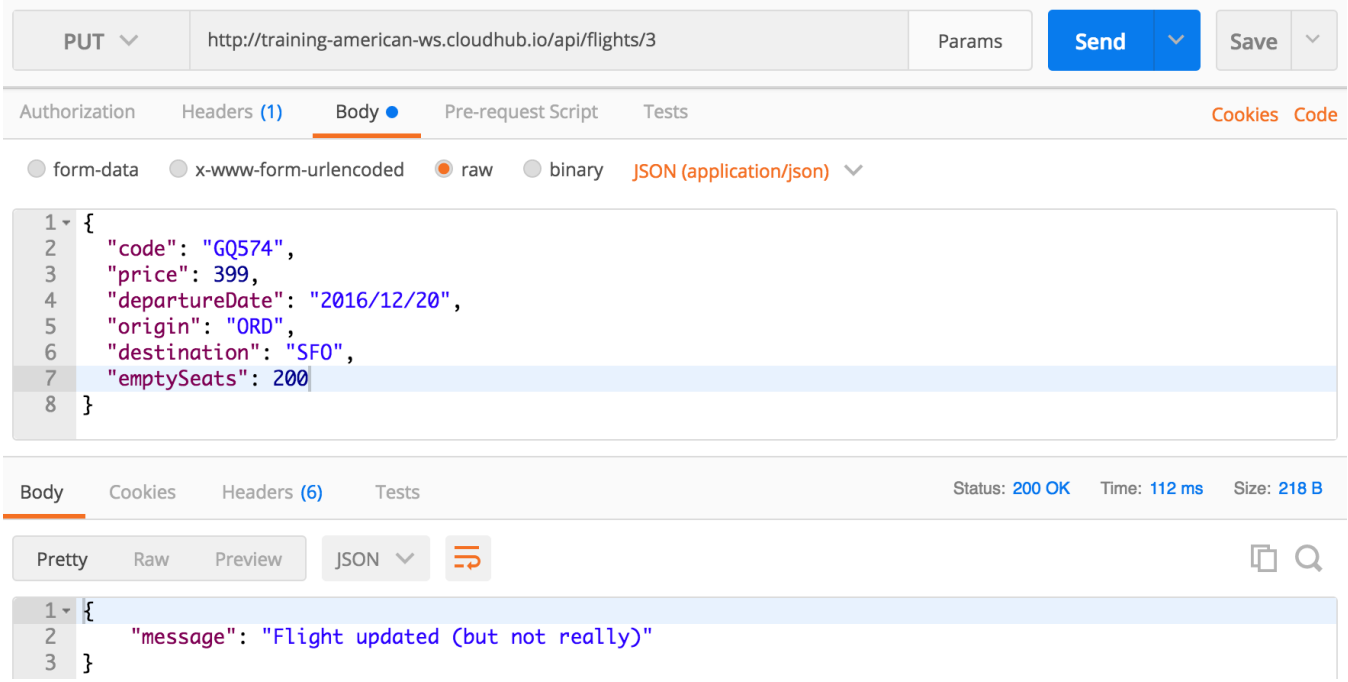
34. Send the request; you should see a 400 response with the message Bad request.



Make a PUT request to update data

35. Change the method to PUT.
36. Add a flight ID to the URL to modify a particular flight.
37. Click the Send button; you should get a bad request message.
38. In the request body field, press Cmd+Z or Ctrl+Z so the emptySeats field is added back.

39. Send the request; you should see the response Flight updated (but not really).

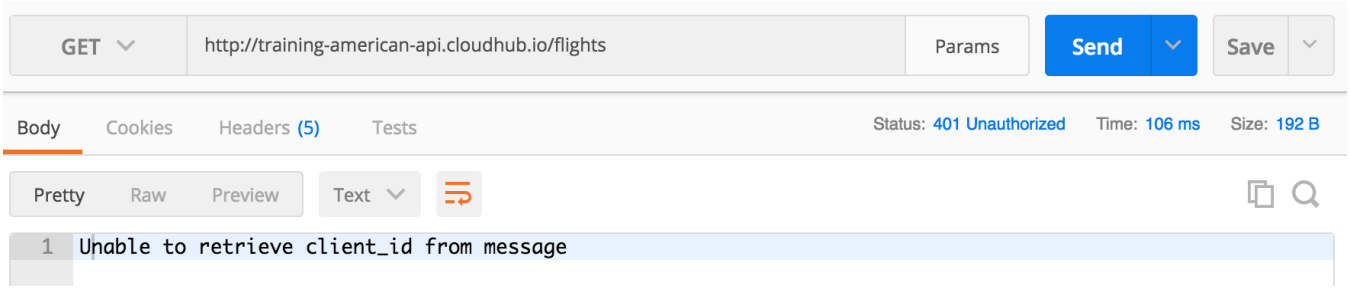


Make a request to a secured API

40. Change the method to GET.
41. Change the request URL to http://training-american-api.cloudhub.io/flights/3.

    Note: The -ws in the URL has been changed to -api and the /api removed.

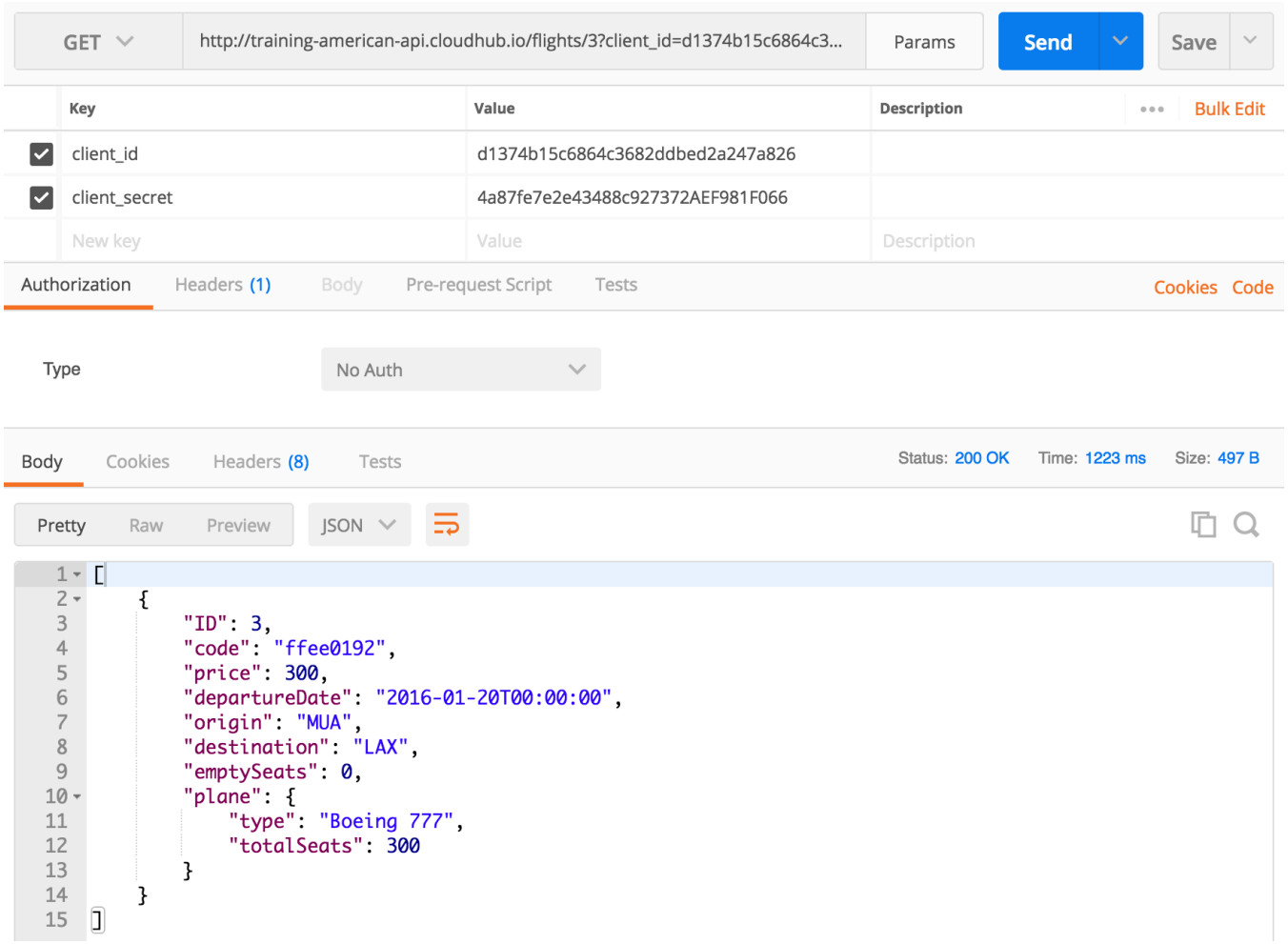
42. Click the Send button; you should get a message about a missing client_id.



43. Return to the course snippets.txt file and copy the value for the American Flights API client_id.
44. Return to Postman and add a request parameter called client_id.
45. Set client_id to the value you copied from the snippets.txt file.
46. Return to the course snippets.txt file and copy the value for the American Flights API client_secret.
47. Return to Postman and add a request parameter called client_secret.

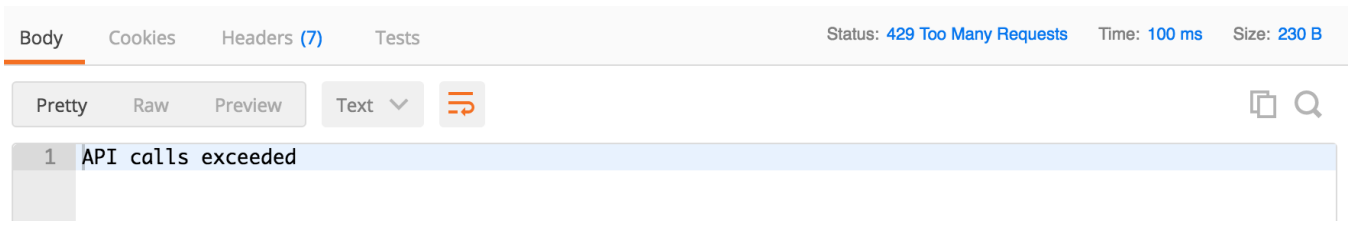
48. Set client_secret to the value you copied from the snippets.txt file.
49. Click the Send button; you should get data for flight 3 again.

    Note: The API service level agreement (SLA) for the application with this client ID and secret has been set to allow three API calls per minute.

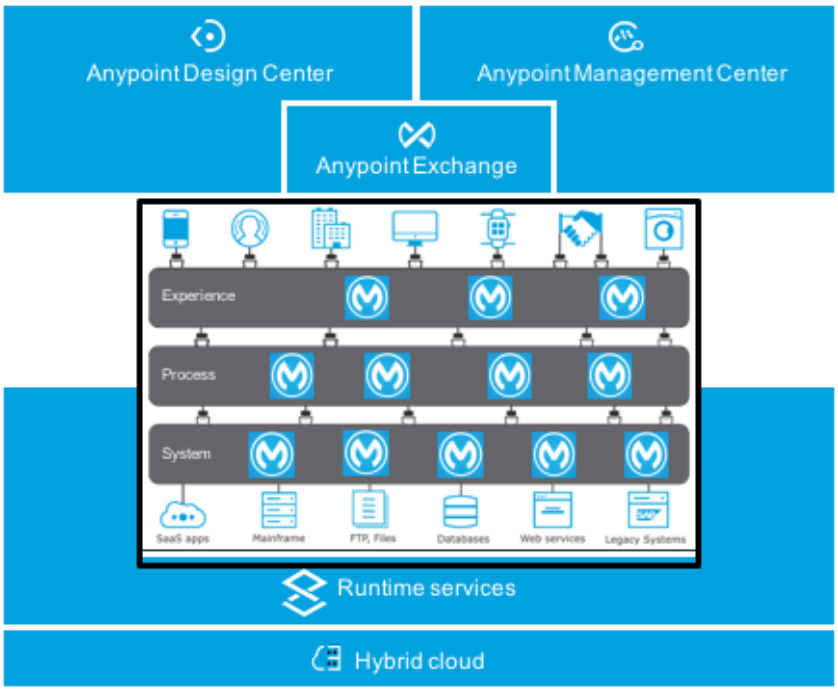


50. Click the Send button three more times; you should get a 429 response and an API calls exceeded message.

    Note: The API service level agreement (SLA) for the application with this client ID and secret has been set to allow three API calls per minute.



Module 2: Introducing Anypoint Platform



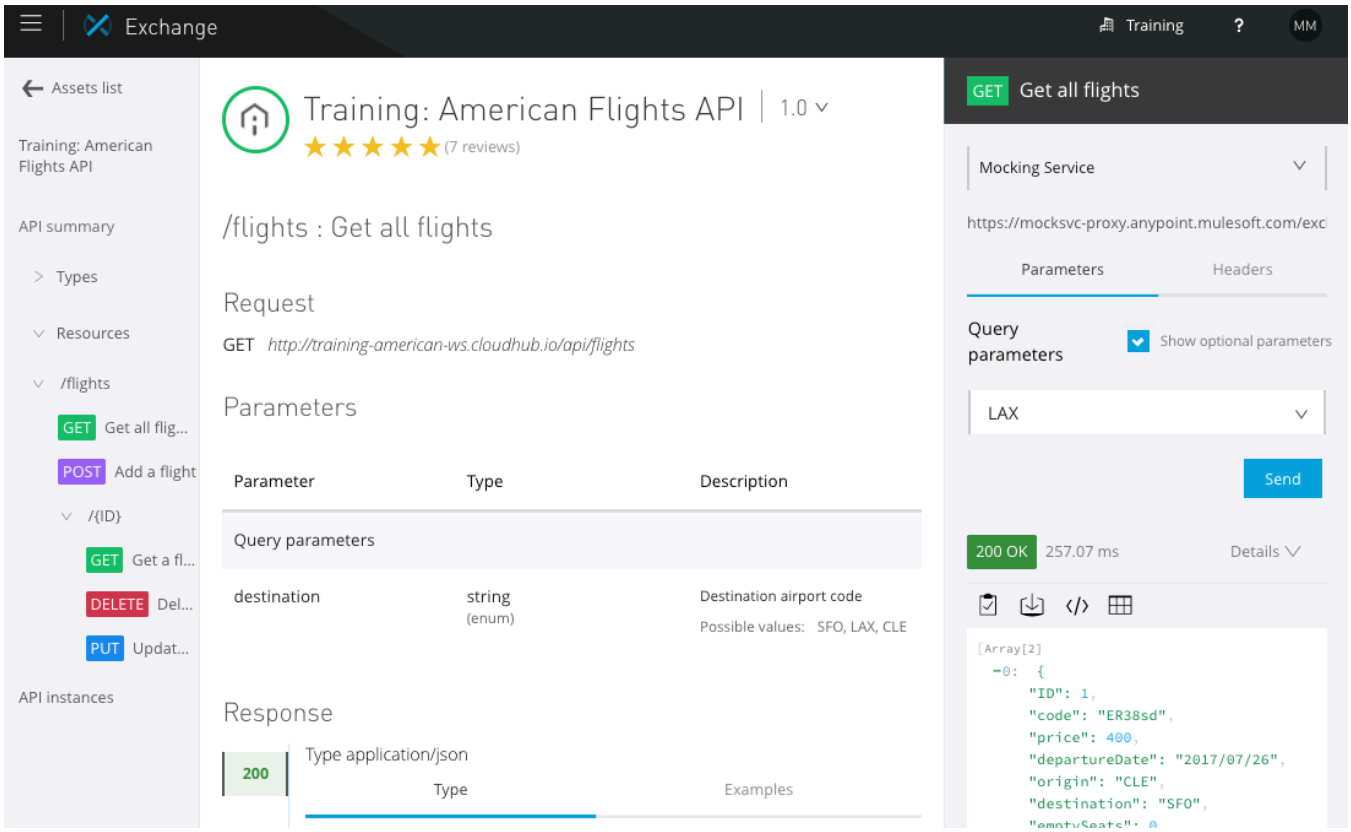
At the end of this module, you should be able to:

- Identify all the components of Anypoint Platform.
- Describe the role of each component in building application networks.
- Navigate Anypoint Platform.
- Locate APIs and other assets needed to build integrations and APIs in Anypoint Exchange.
- Build basic integrations to connect systems using flow designer.

Walkthrough 2-1: Explore Anypoint Platform and Anypoint Exchange

In this walkthrough, you get familiar with the Anypoint Platform web application. You will:

- Explore Anypoint Platform.
- Browse Anypoint Exchange.
- Review an API portal for a REST API in Exchange.
- Discover and make calls to the Training: American Flights API.



Return to Anypoint Platform

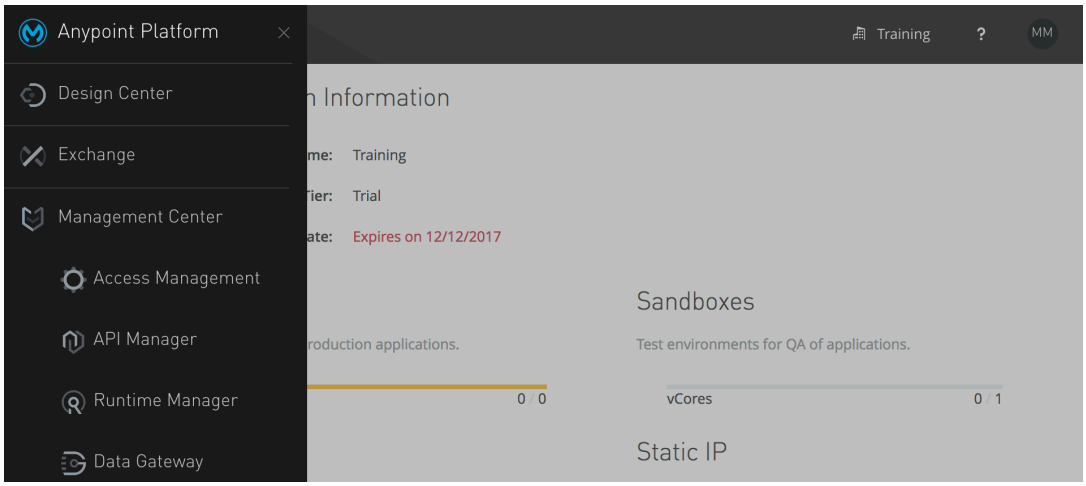
1. Return to Anypoint Platform in a web browser.

   Note: If you closed the browser window or logged out, return to https://anypoint.mulesoft.com and log in.

2. Click the menu button located in the upper-left in the main menu bar.

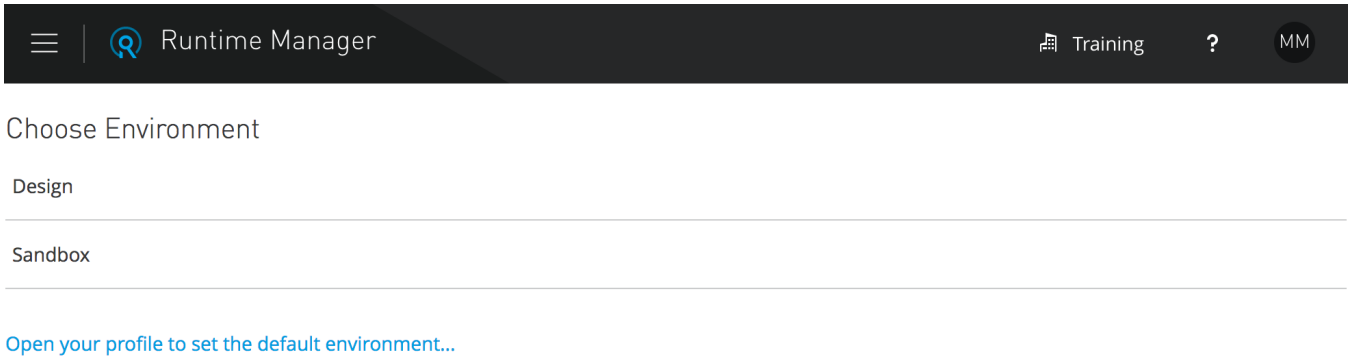
3. In the menu that appears, select Anypoint Platform; this will return you to the home page.

   Note: This will be called the main menu from now on.



Explore Anypoint Platform

4. In the main menu, select Access Management.
5. In the main menu, select Design Center.
6. In the main menu, select Runtime Manager.
7. If you get a Choose environment page, select Design.

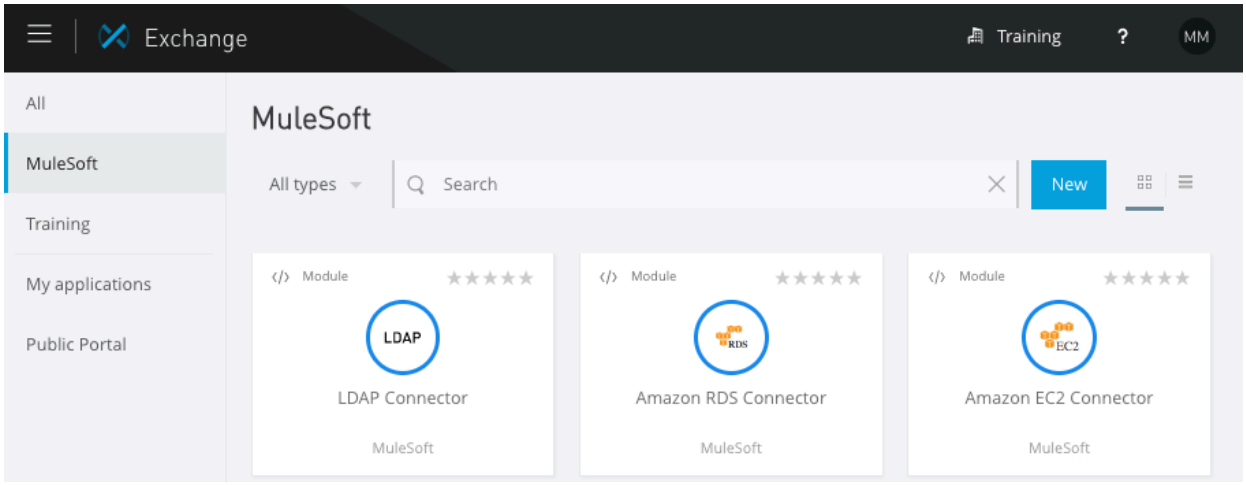


8. In the main menu, select API Manager.

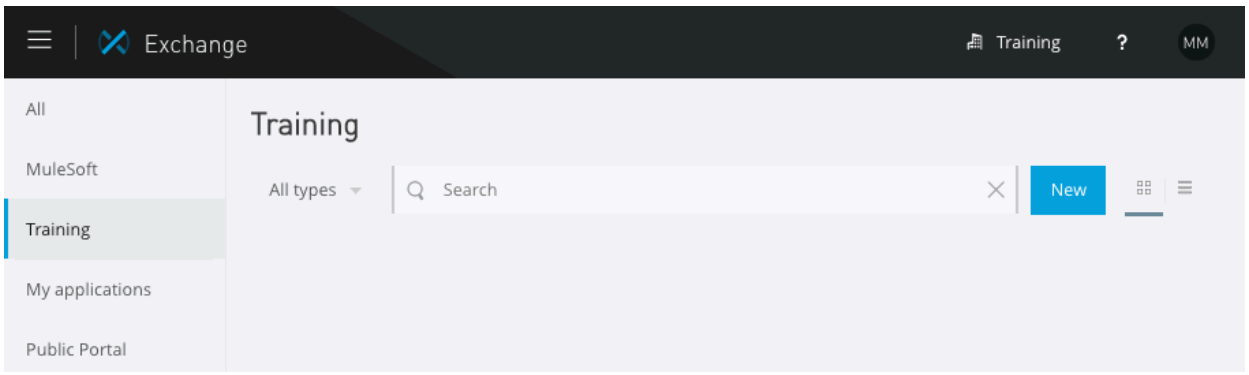
Explore Anypoint Exchange

9. In the main menu, select Exchange.

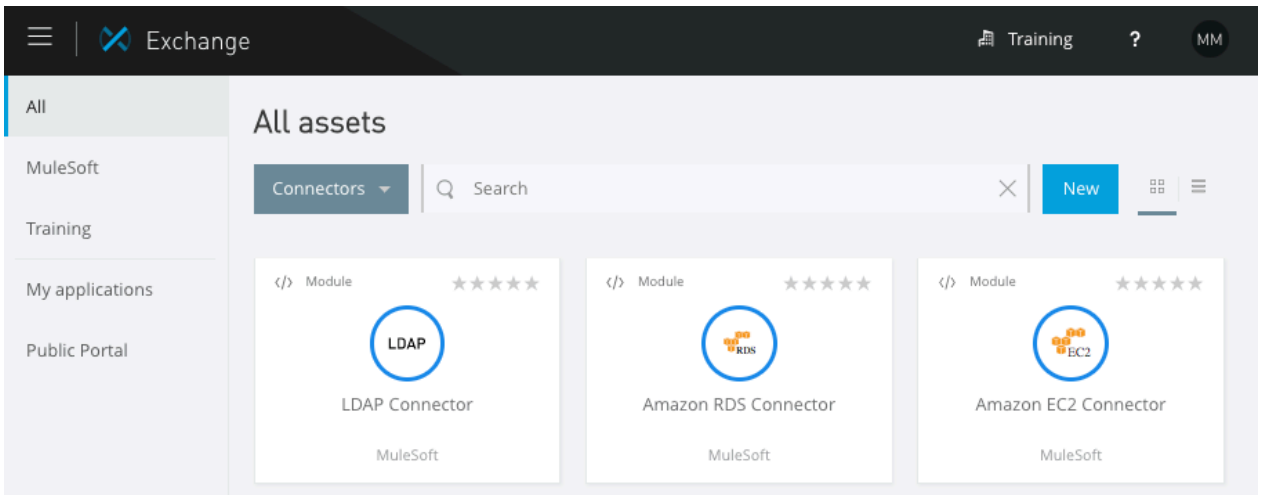
10. In the left-side navigation, click MuleSoft; you should see all the content in the public Exchange.



11. In the left-side navigation, click the name of your organization beneath MuleSoft (Training in the screenshots); you should now see only the content in your private Exchange, which is currently empty.

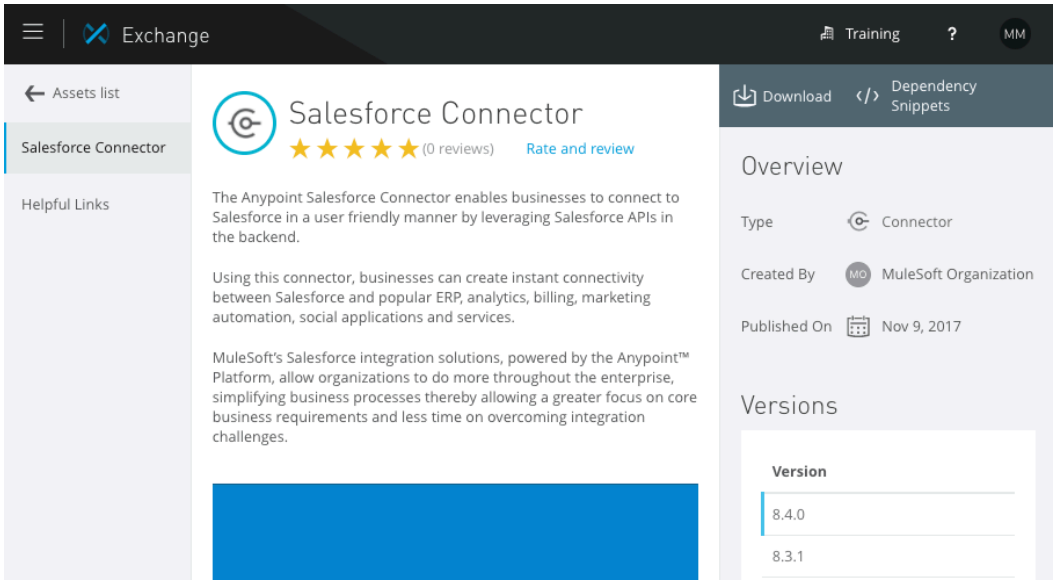


12. In the left-side navigation, click All.
13. In the types menu, select Connectors.



14. Click one of the connectors and review its information.
15. In the left-side navigation, click the Assets list link.
16. Select the Salesforce connector (or search for it if it is not shown) and review its details.

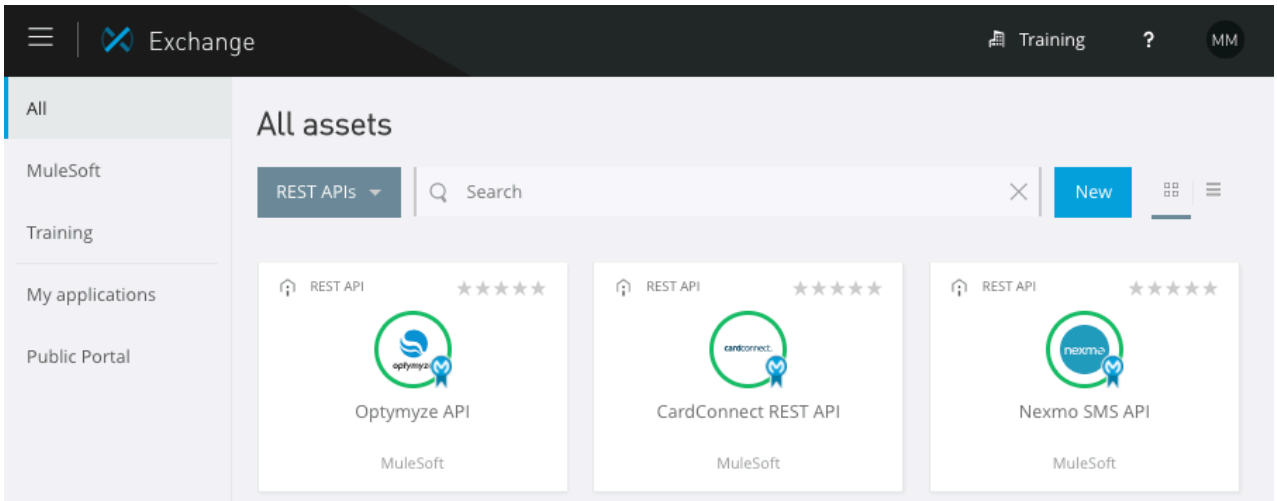
    Note: The Salesforce connector is used in the Development Fundamentals course.



17. In the left-side navigation, click Assets list.
18. In the types menu, select Templates (and remove anything from the search field).

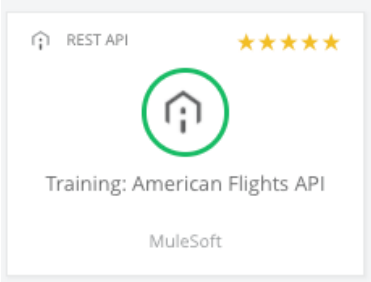
Browse REST APIs in Anypoint Exchange

19. In the types menu, select REST APIs.
20. Browse the APIs.

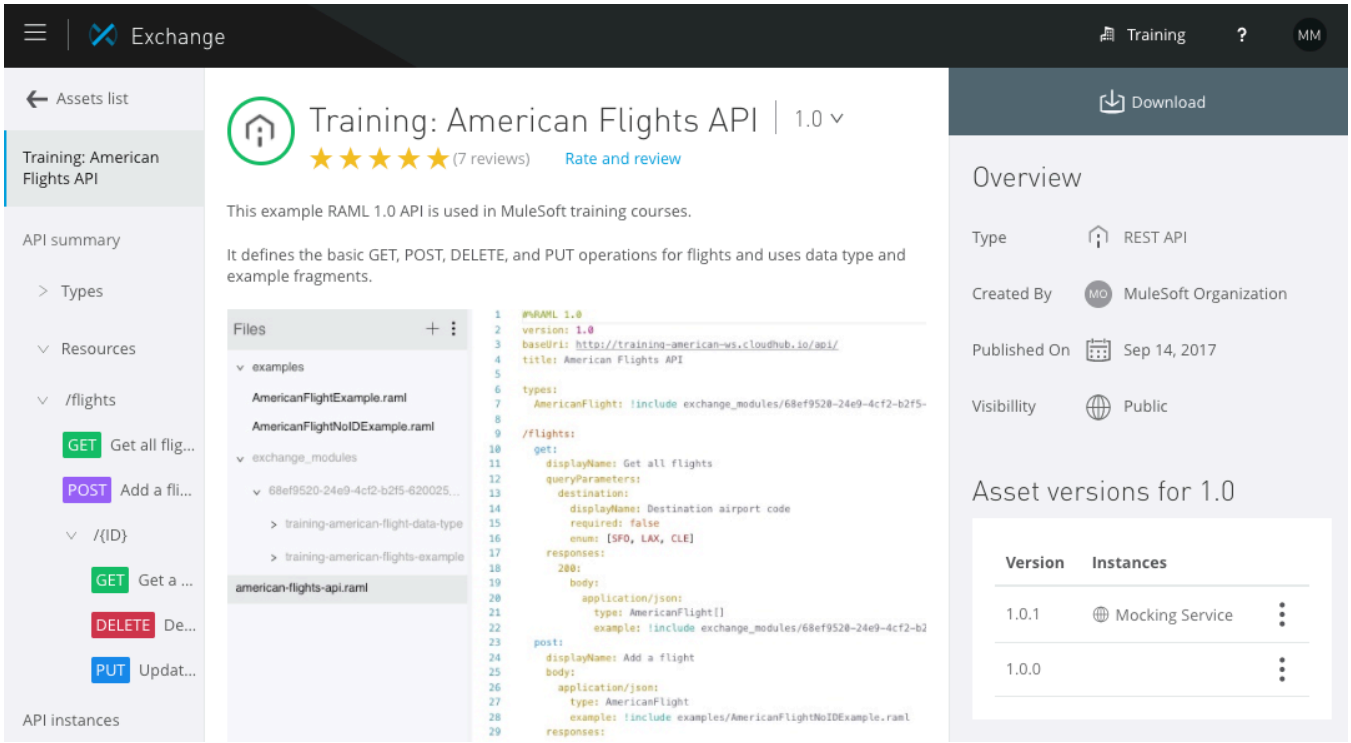


Discover and review the API portal for the Training: American Flights API

21. Locate and click the Training: American Flights API.

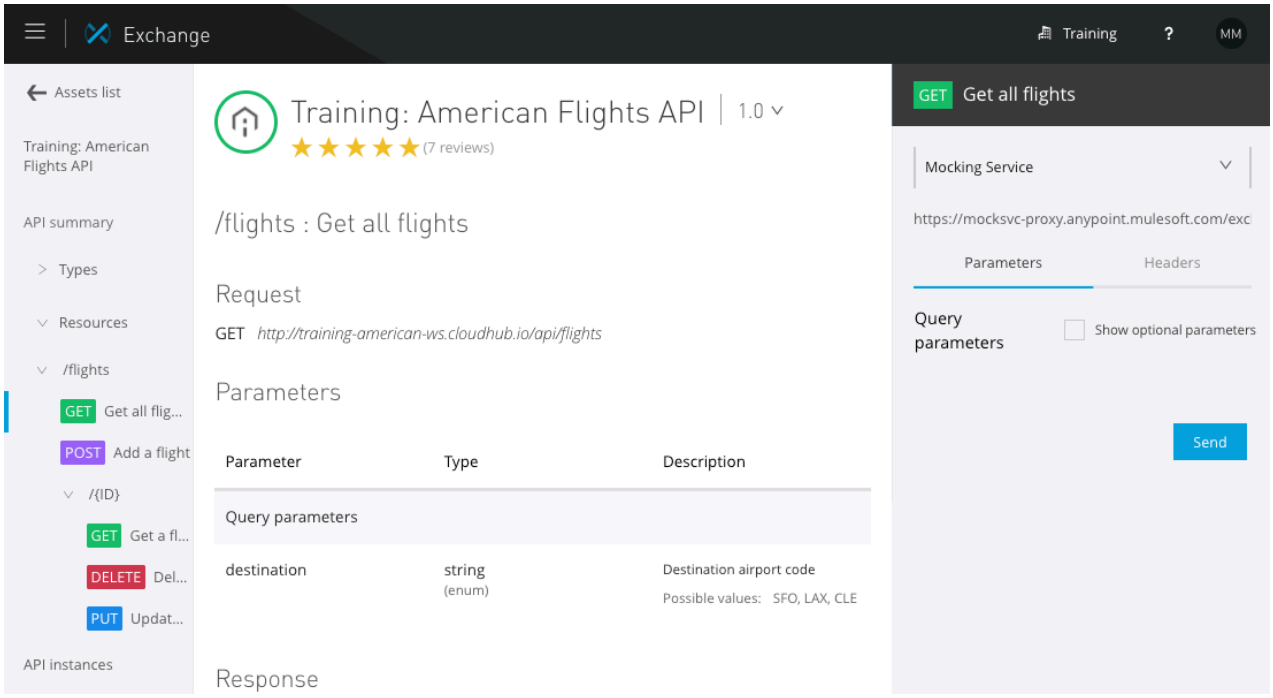


22. Review the API portal.

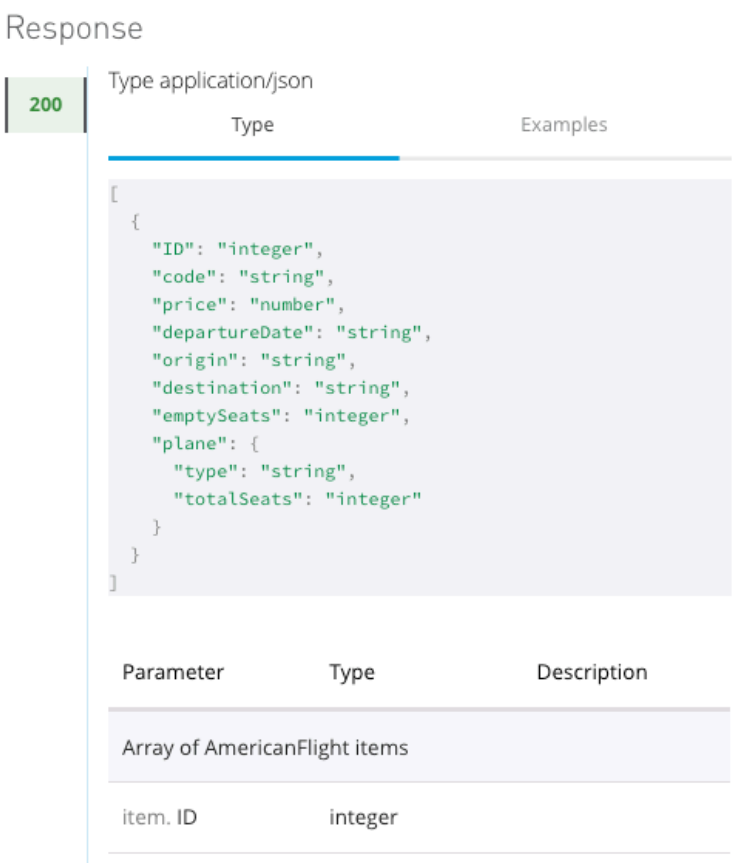


23. In the left-side navigation, expand and review the list of available resources.
24. Click the GET link for the /flights resource.

25. On the /flights: Get all flights page, review the information for the optional query parameter.



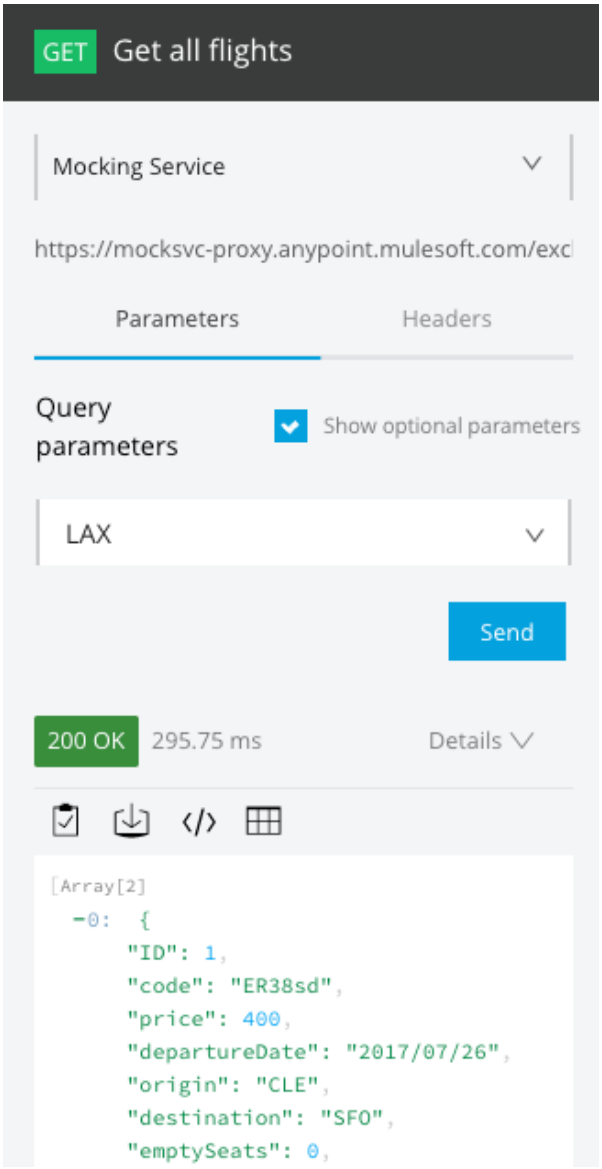
26. Scroll down and locate the type and details for the data returned from a call.



27. Select the Examples tab; you should see an example response.

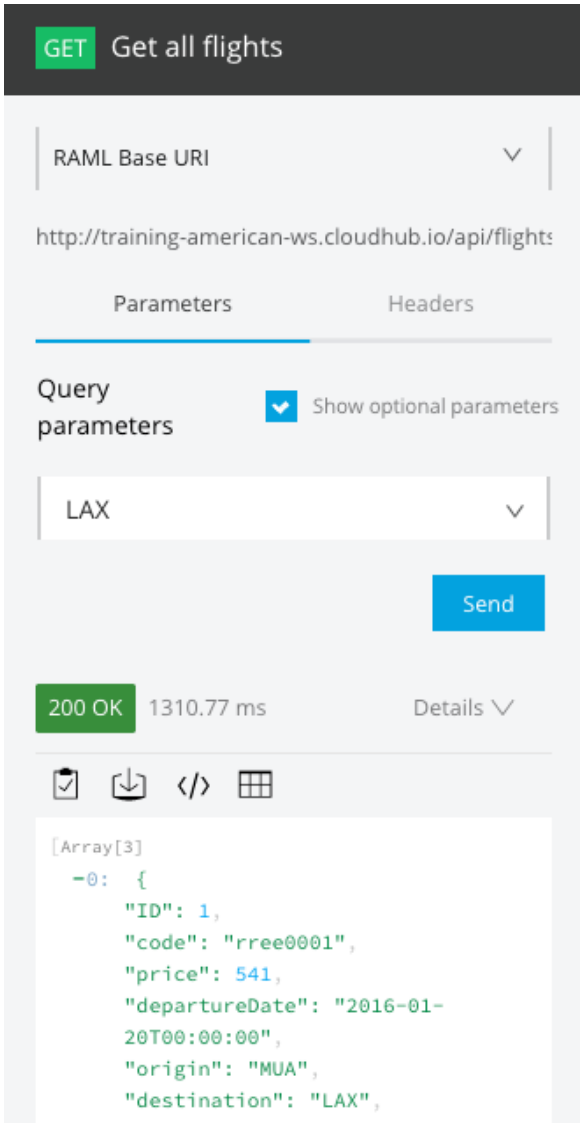
Use the API console to make calls to the Training: American Flights API

28. Scroll up and locate the API console on the right-side on the page.
29. Select to show optional query parameters and select a value.
30. Select a destination and click Send; you should get the example data returned using the mocking service.



31. In the API console, change the API instance from Mocking Service to RAML Base URI.

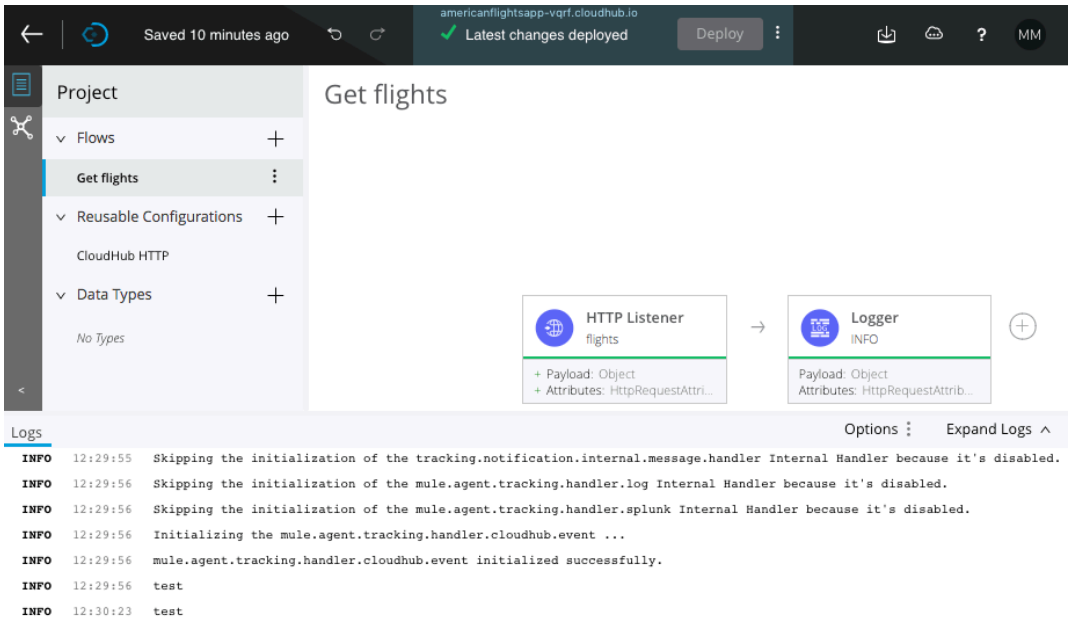
32. Click Send again; you should get results from the actual API implementation for the destination you selected.



Walkthrough 2-2: Create a Mule application with flow designer

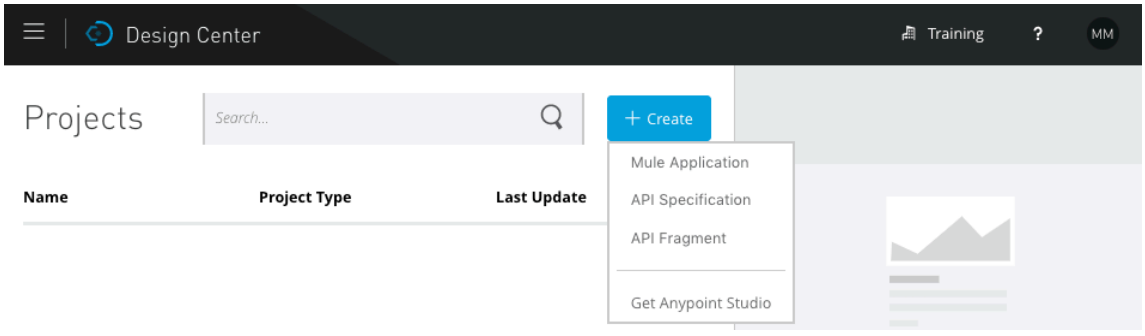
In this walkthrough, you build, run, and test a basic Mule application with flow designer. You will:

- Create a new Mule application project in Design Center.
- Create an HTTP trigger for a flow in the application.
- Add a Logger component.
- Deploy, run, and test the application.
- View application information in Runtime Manager.



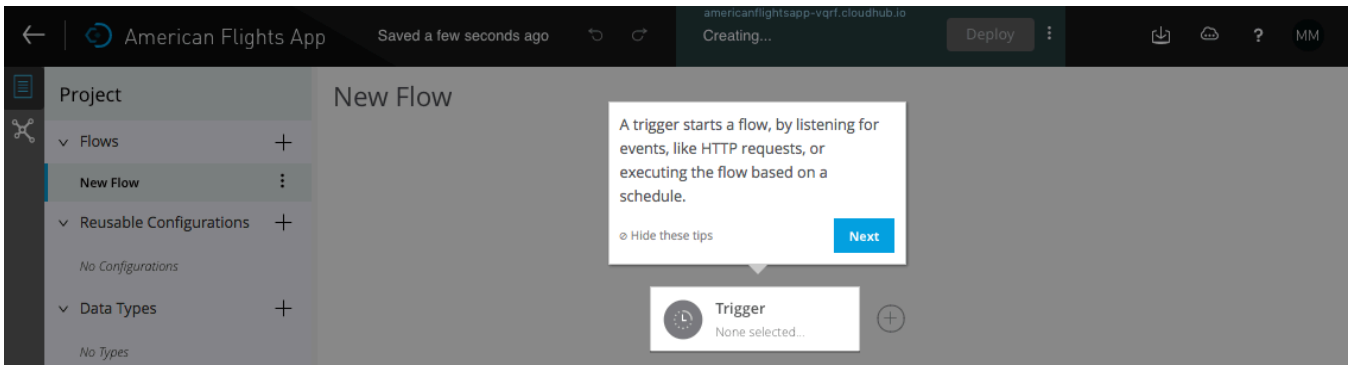
Create a Mule application project in Design Center

1. Return to Anypoint Platform.
2. In the main menu, select Design Center.
3. Click the Create button and select Mule Application.

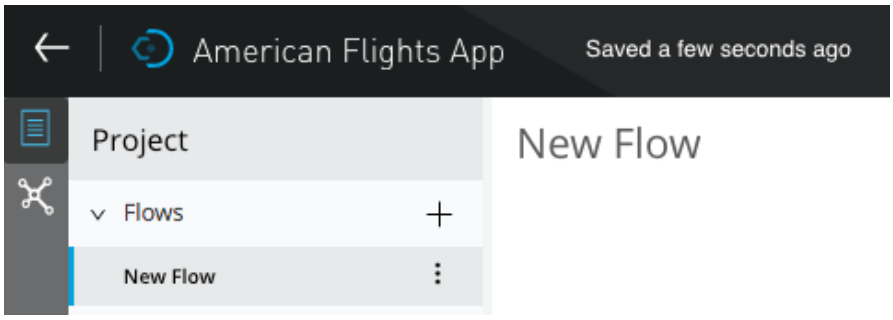


4. In the New Mule Application dialog box, set the project name to American Flights App.

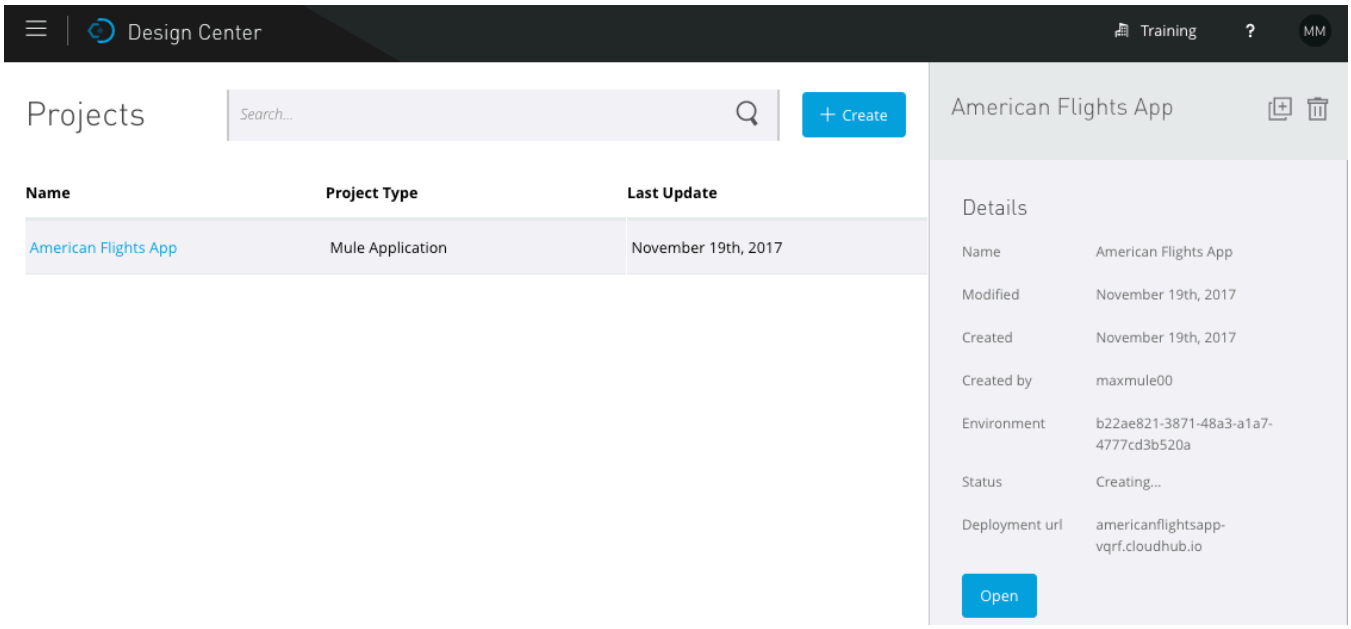
5. Click Create; flow designer should open.



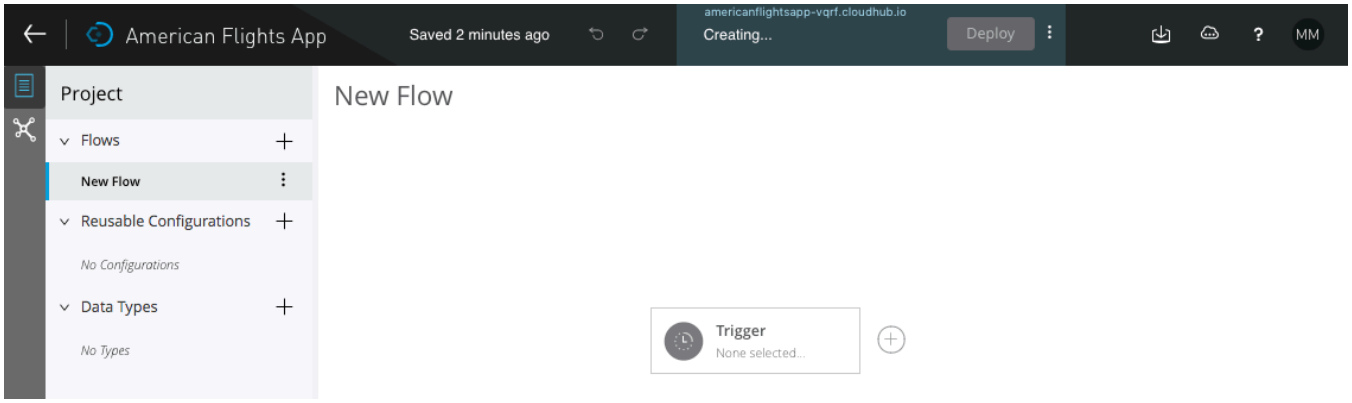
6. In the pop-up box in flow designer, click Hide these tips.
7. Click the arrow icon in the upper-left corner; you should return to the Design Center.



8. In the Design Center project list, click the row containing the American Flights App; you should see information about the project displayed on the right side of the page.

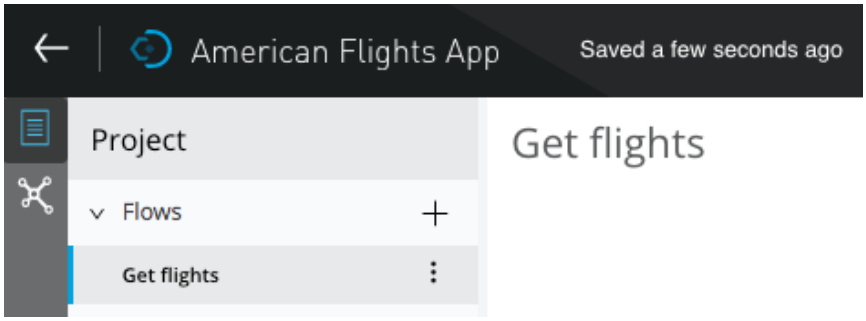


9. Click the Open button or click the American Flights App link in the project list; the project should open in flow designer.



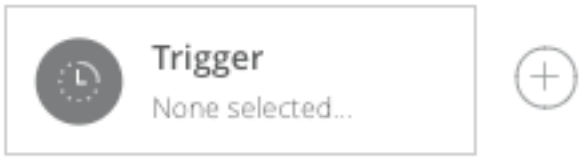
Rename the flow

10. Locate New Flow in the project explorer.
11. Click its option menu and select Rename.
12. In the Rename Flow dialog box, set the name to Get flights and click OK.

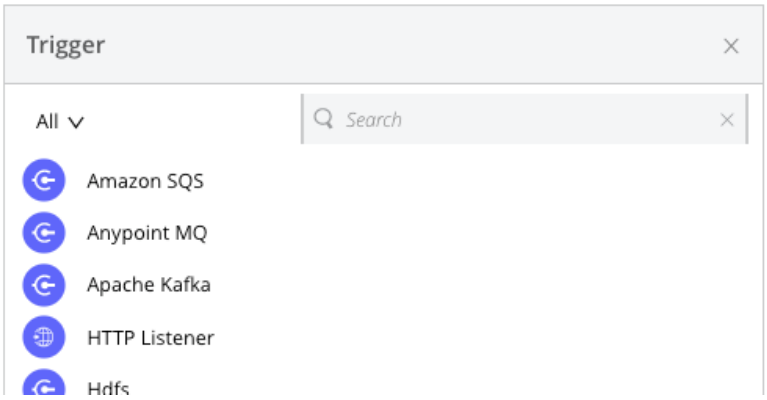


Create an HTTP trigger for a flow in the application

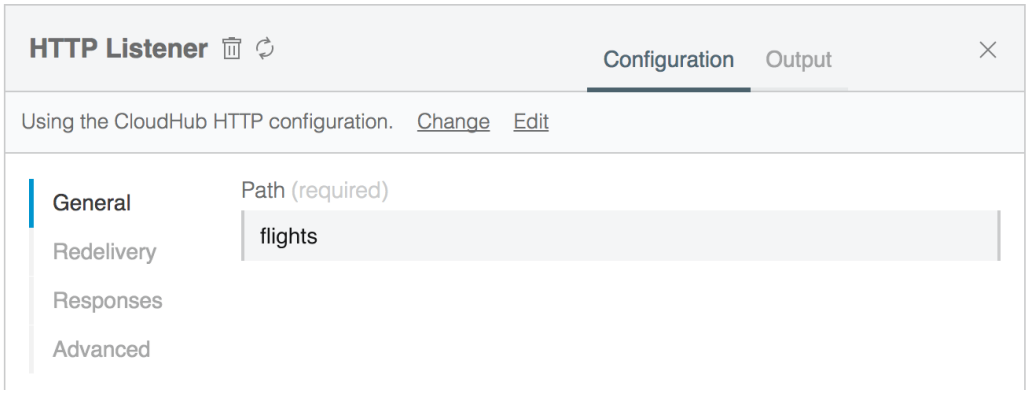
13. In flow designer, click the Trigger card.



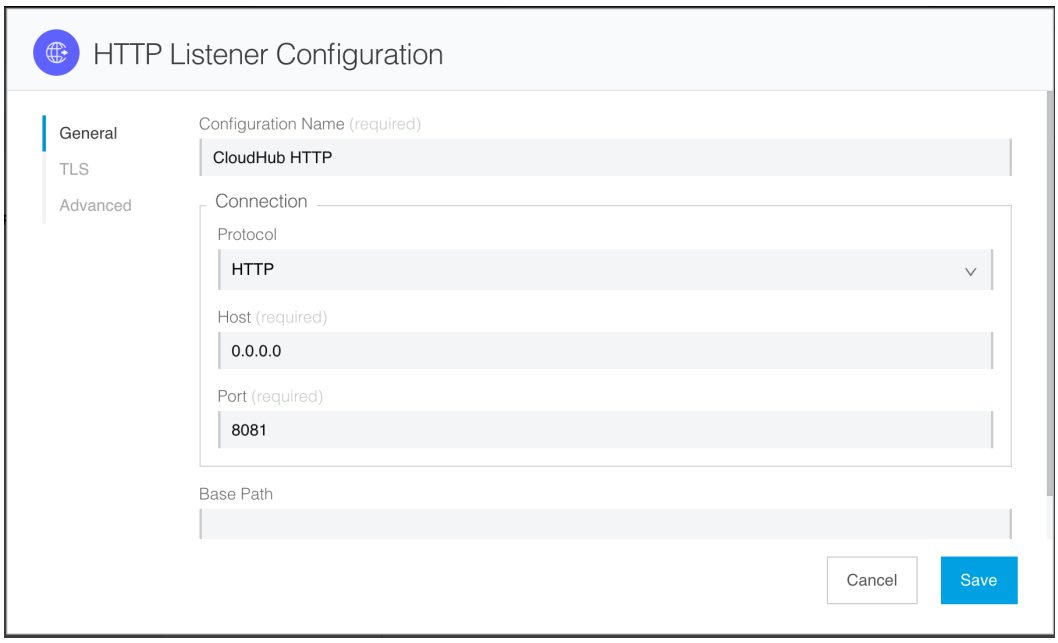
14. In the Trigger card, select HTTP Listener.



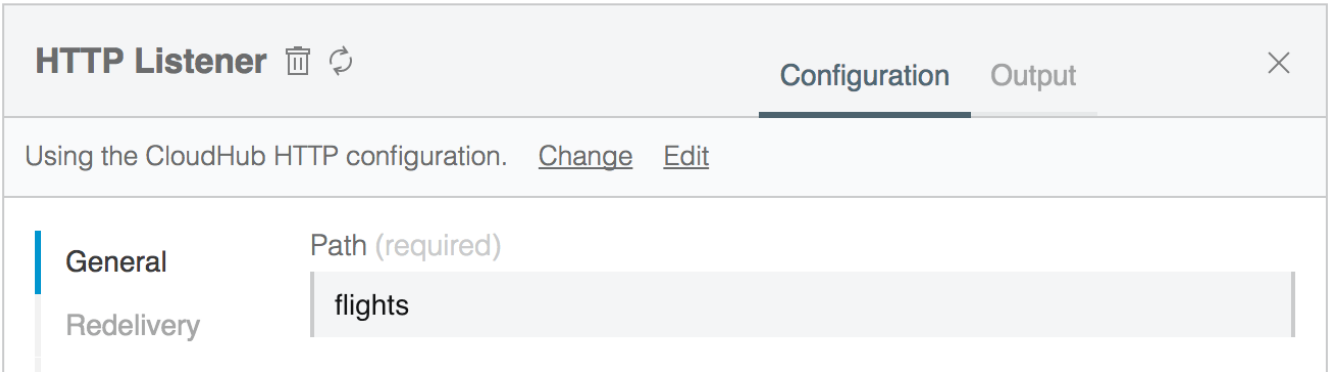
15. In the HTTP Listener dialog box, set the path to flights.



16. Click the Edit link for the CloudHub HTTP configuration.
17. In the HTTP Listener Configuration dialog box, review the information and click Cancel.

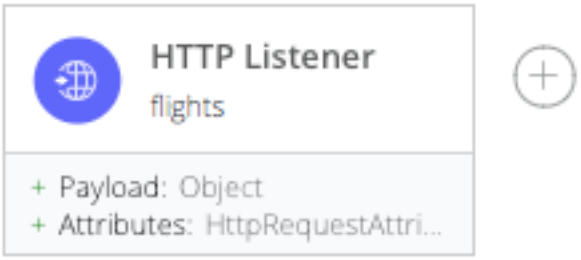


18. In the HTTP Listener dialog box, click the close button in the upper-right corner.

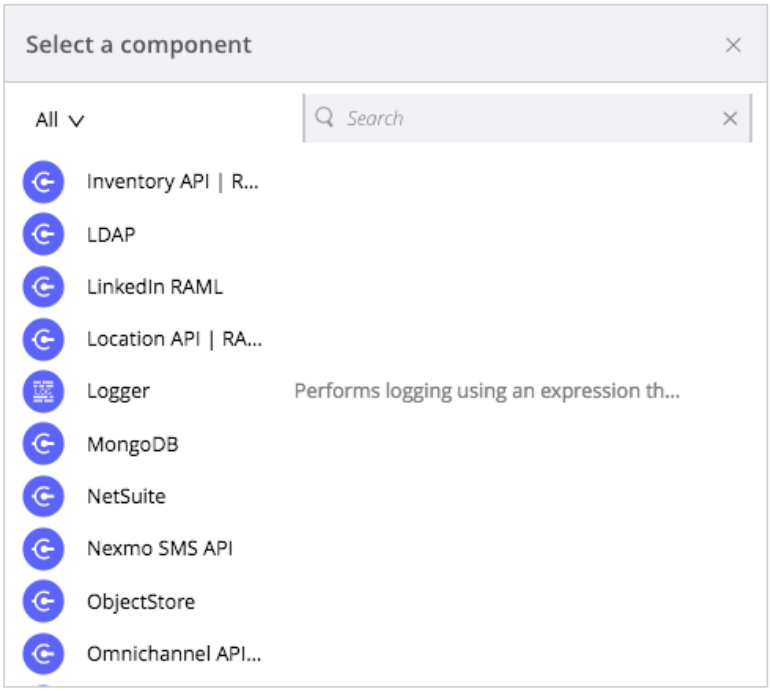


Add a Logger

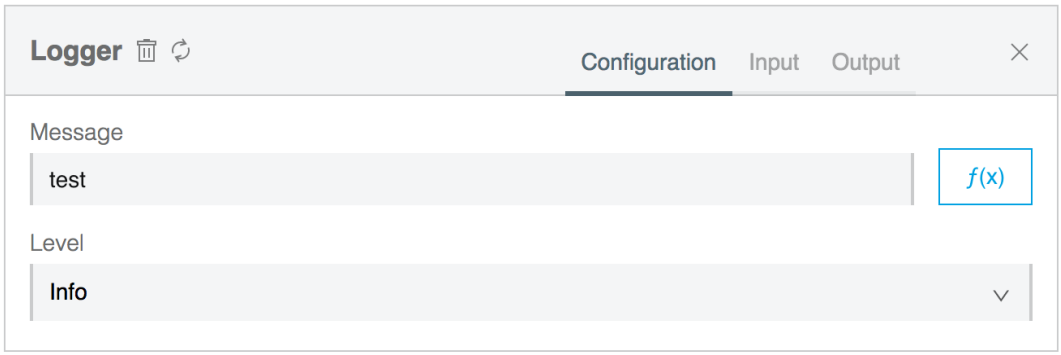
19. Click the add button next to the HTTP Listener card.



20. In the Select a component dialog box, select Logger.



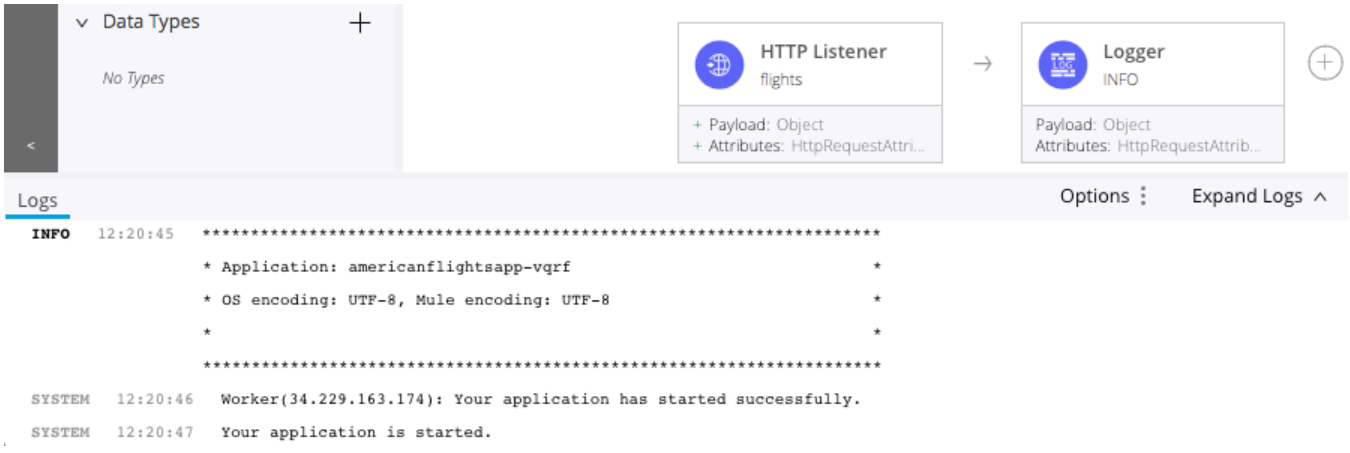
21. In the Logger dialog box, set the message to test.



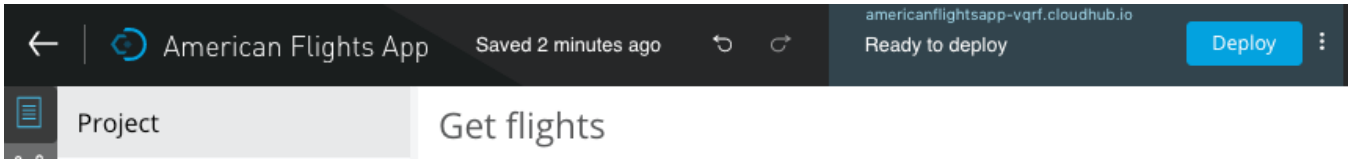
22. Close the card.
23. Notice that there are gray lines across both cards.

Deploy the application

24. Click the Logs tab located in the lower-left corner of the window; you should see that your application is already started.



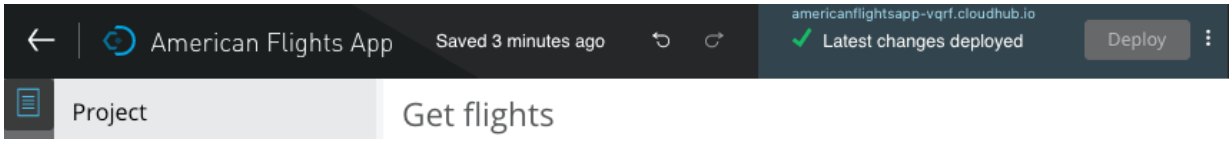
25. Locate the application status in the main menu bar; it should say Ready to deploy.



26. Look at the generated URL for the application.

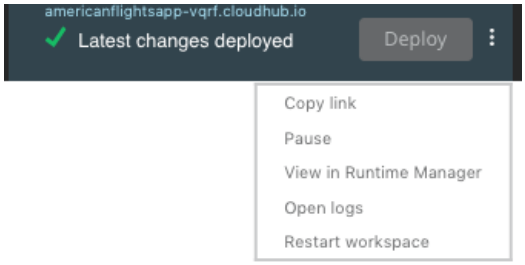
    Note: The application name is appended with a four-letter suffix to guarantee that it is unique across all applications on CloudHub.

27. Click the Deploy button.
28. Watch the application status; it should change from Ready to Deploying then to Latest changes deployed.



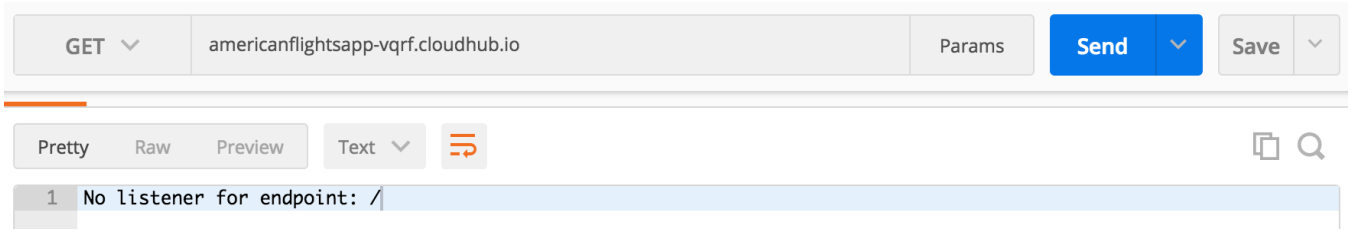
    Note: If your application fails to deploy, look at the messages in the Logs panel. If there is no message about incorrect syntax, try restarting the workspace by clicking the options menu in the application status area and selecting Restart workspace.

29. Click the options menu in the application status area and select Copy link.

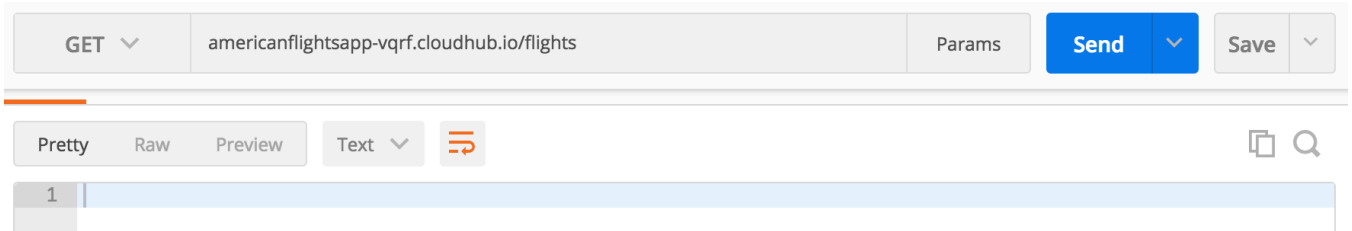


Test the application

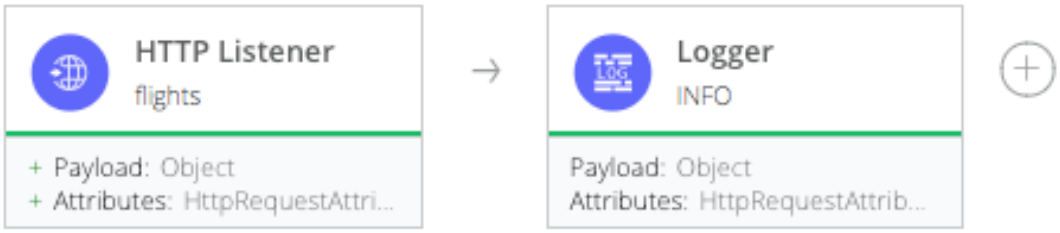
30. Return to Postman, paste the copied link, and click Send; you should get a No listener for endpoint: / message.



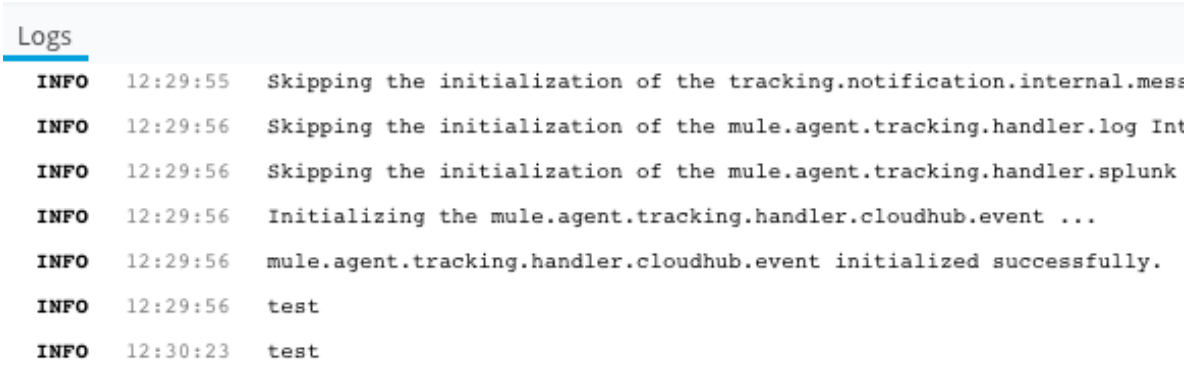
31. Add /flights to the path and click Send; you should get a 200 response with no body.



32. Click Send again to make a second request.
33. Return to flow designer.
34. Notice that there are now green lines across both cards.

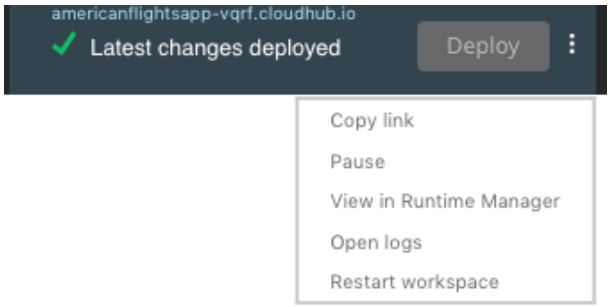


35. Look at the logs; you should see your Logger message displayed twice.

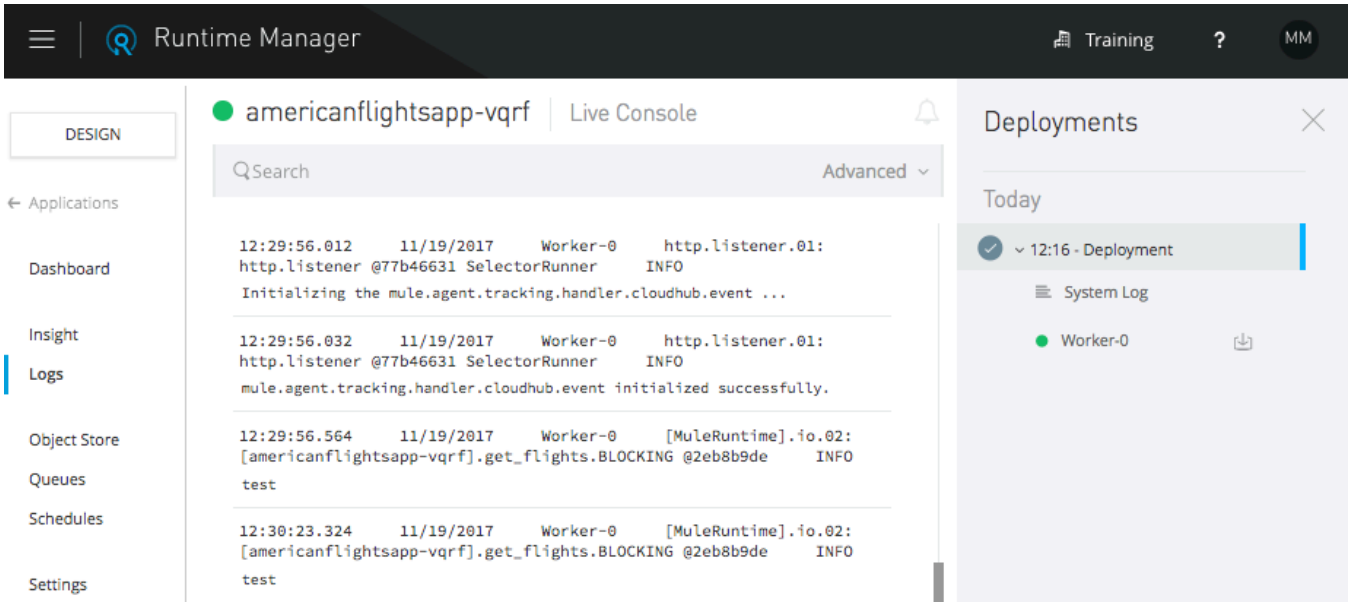


View the application in Runtime Manager

36. Click the options menu in the application status area and select View in Runtime Manager; Runtime Manager should open in a new tab.

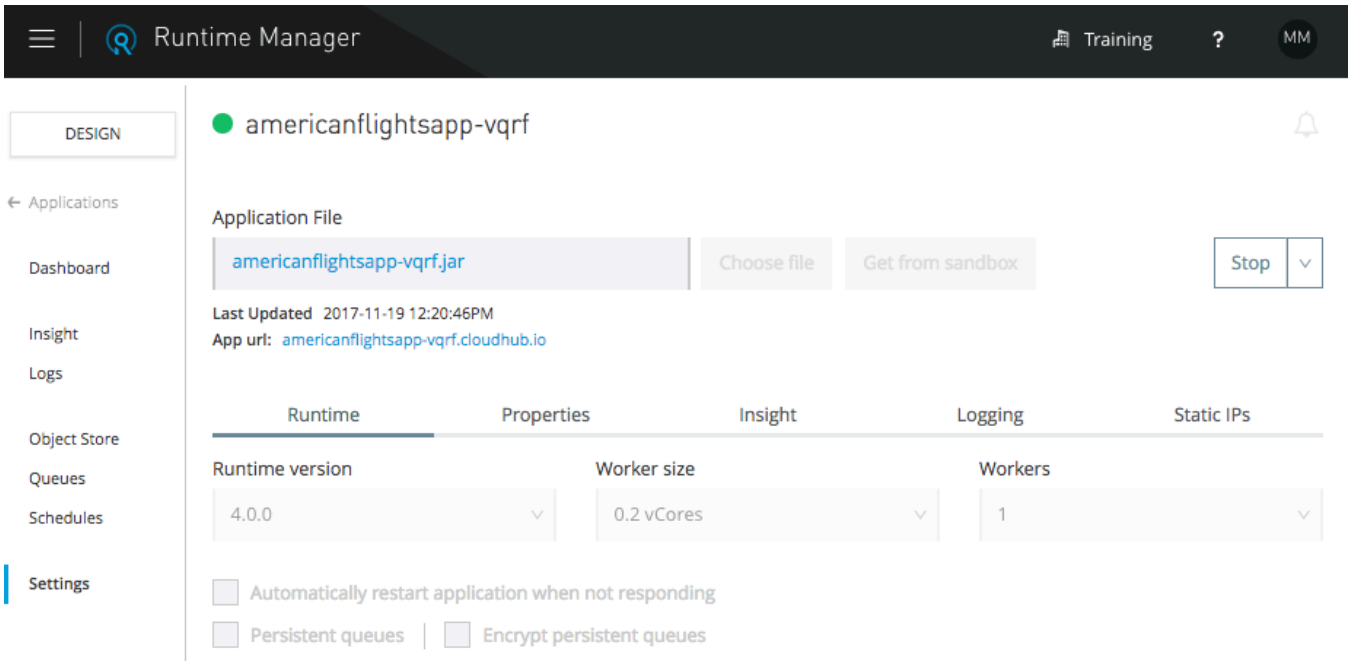


37. In the new browser tab that opens with Runtime Manager, review the application log file; you should see your test log messages.



38. In the left-side navigation, click Settings.
39. Review the settings page and locate the following information for the application:

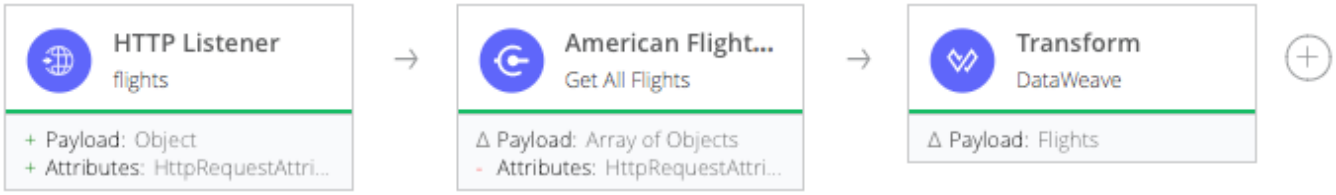
- To which environment it was deployed
- To what type of Mule runtime it was deployed
- To what size worker it was deployed



Walkthrough 2-3: Create an integration application with flow designer that consumes an API

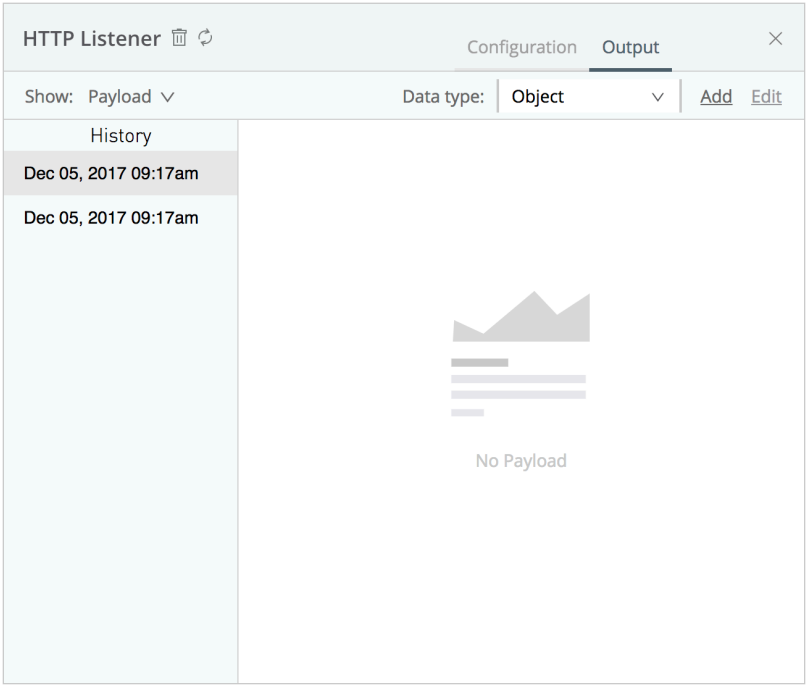
In this walkthrough, you build an integration application to consume an API from Anypoint Exchange. You will:

- Examine Mule event data for calls to an application.
- Use the American Flights API in Anypoint Exchange to get all flights.
- Transform data returned from an API to another format.

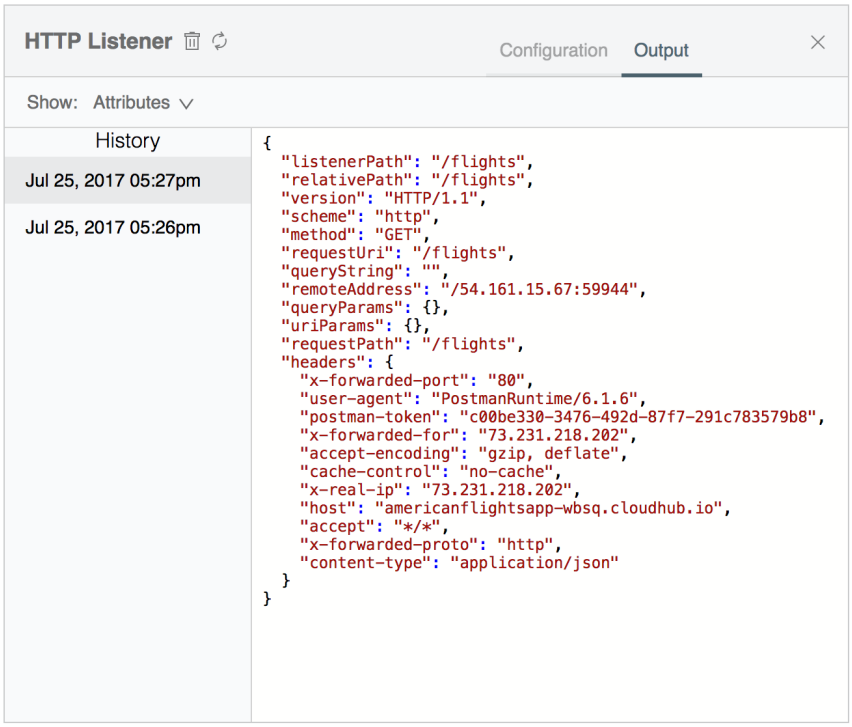


Review Mule event data for the calls to the application

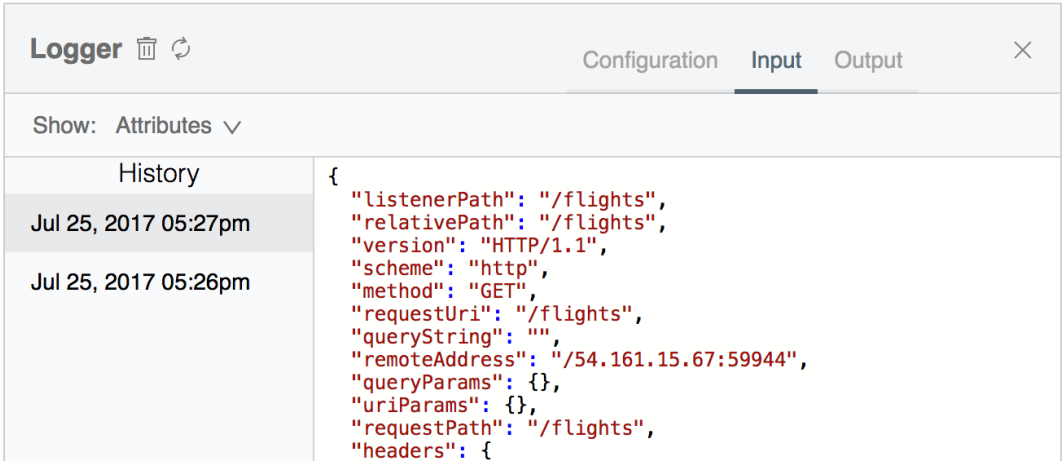
1. Return to the American Flights App in flow designer.
2. Click the title bar of the Logs panel to close it.
3. Expand the HTTP Listener card.
4. Click the Output tab.
5. Locate the Show drop-down menu that currently has Payload selected.



6. Locate your two calls to the application in the History panel; there should be no message payload for either call.
7. Change the Show drop-down menu to Attributes.
8. Review the attributes for the Mule event leaving the HTTP Listener processor.



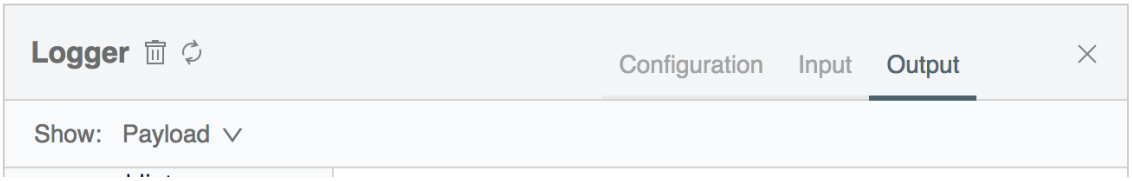
9. Close the card.
10. Open the Logger card.
11. Click the Input tab.
12. Review the payload and attributes values for the two calls.



13. Click the Output tab and review the payload and attributes values for the calls.

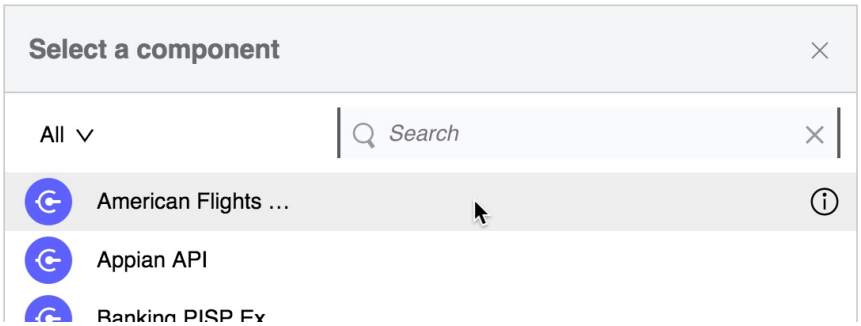
Delete a card

14. Click the Remove button at the top of the card.

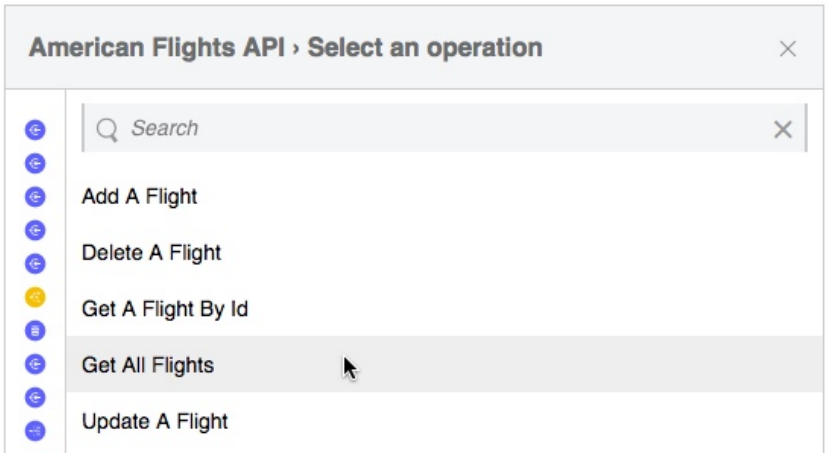


Use the American Flights API in Anypoint Exchange to get all flights

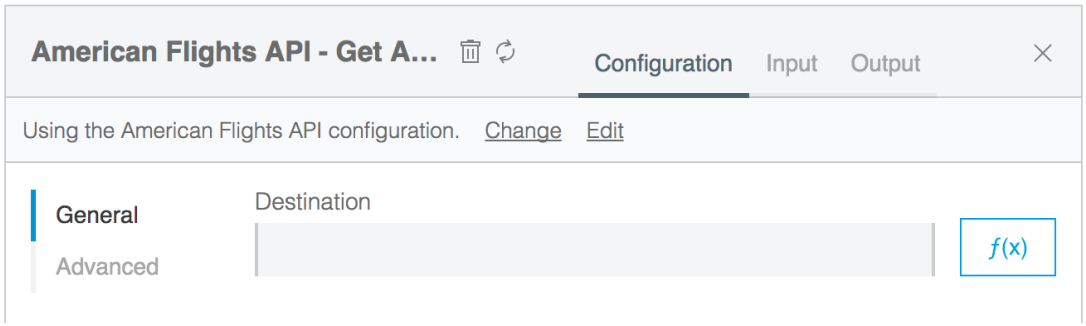
15. Click the Add button next to the HTTP Listener card.
16. In the Select a component dialog box, select the American Flights API.



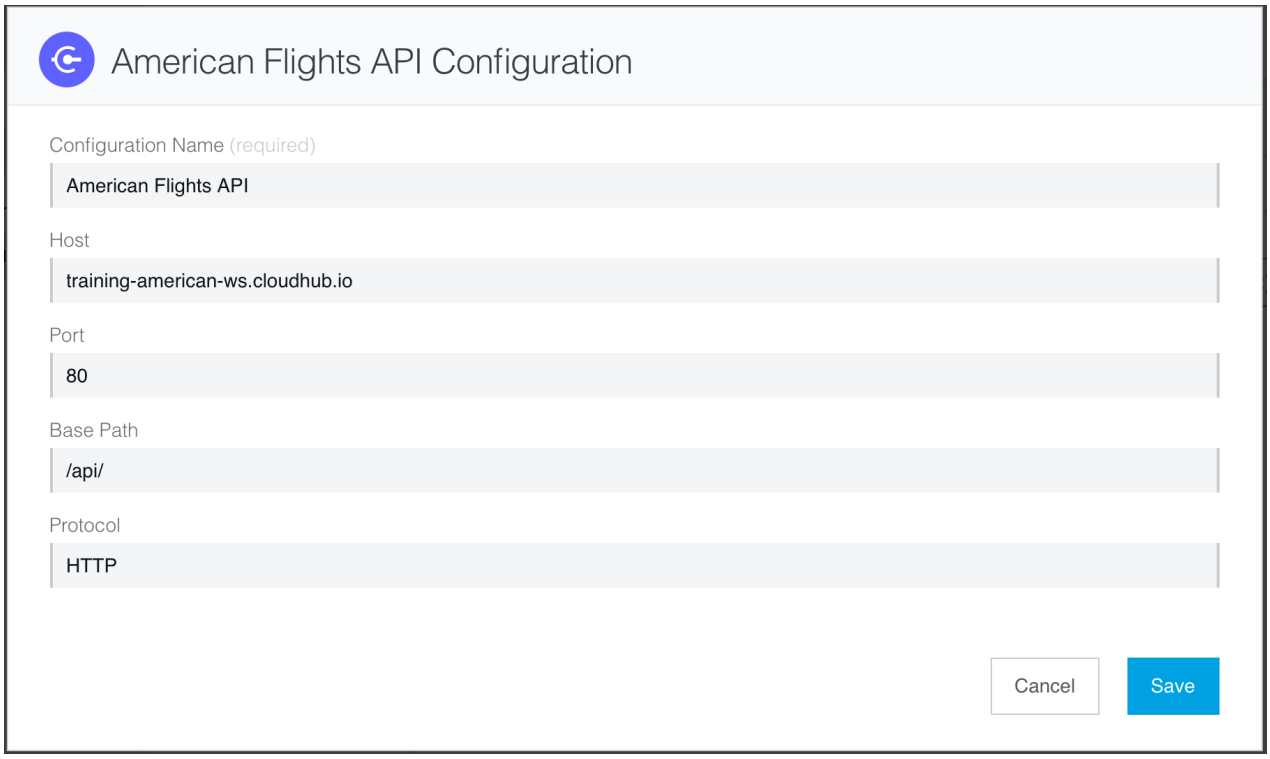
17. In the American Flights API > Select an operation dialog box, select Get All Flights.



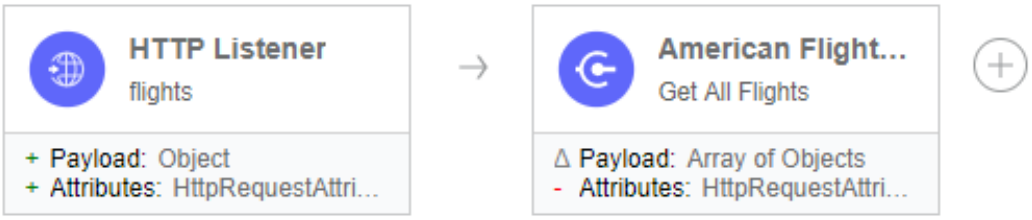
18. In the Get All Flights dialog box, click the Edit link for the American Flights API configuration.



19. Review the information and click Cancel.



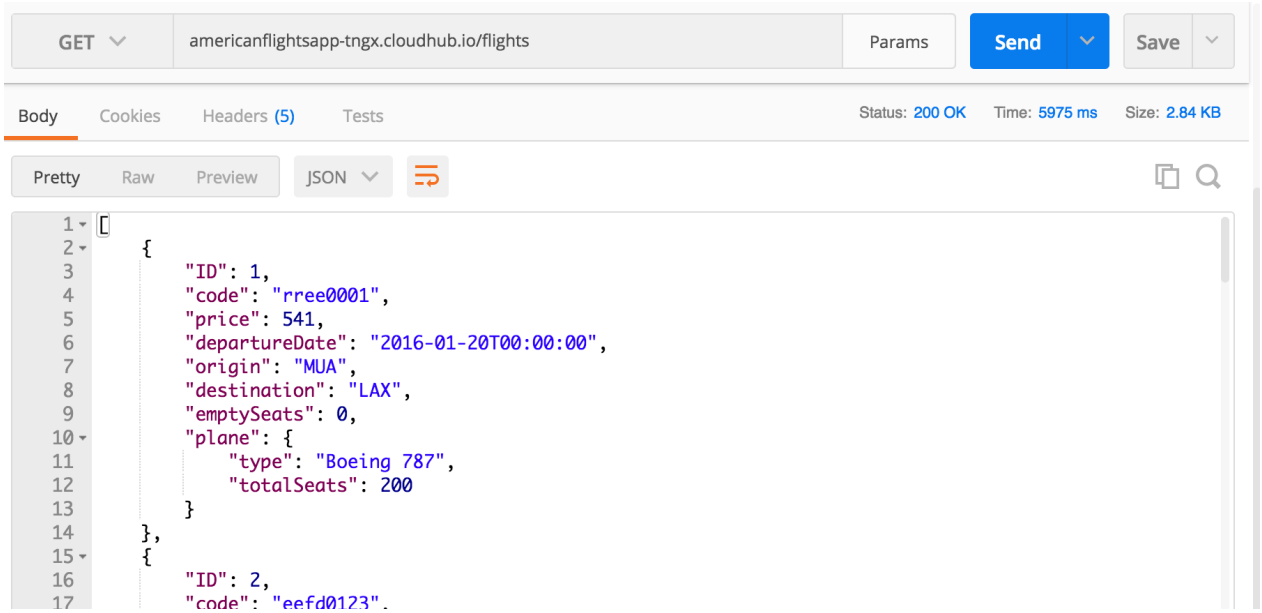
20. Close the American Flights API card.



21. Click the Deploy button in the main menu bar.

Test the application

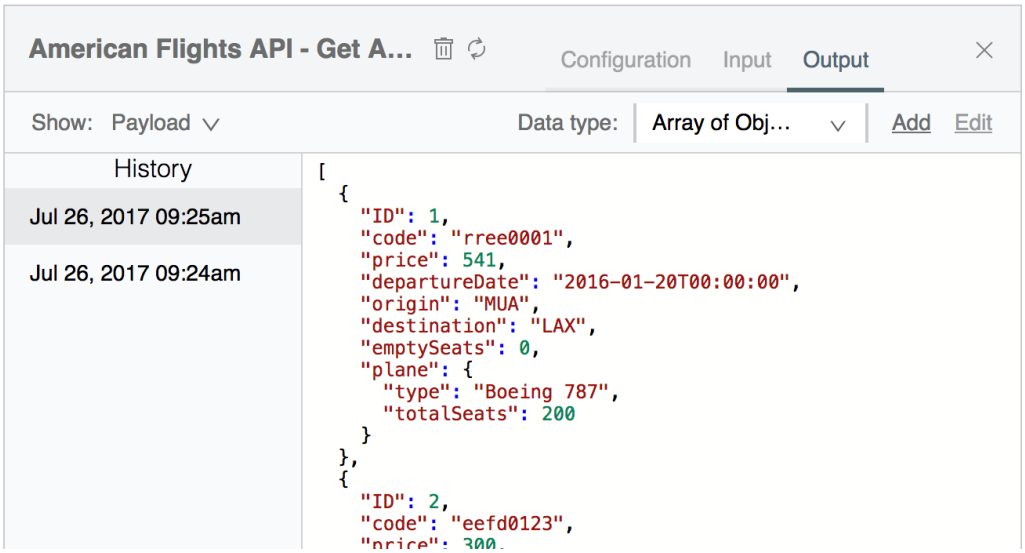
22. Return to Postman and click Send; you should see flight data.



23. Click Send again to make a second request.

Review Mule event data

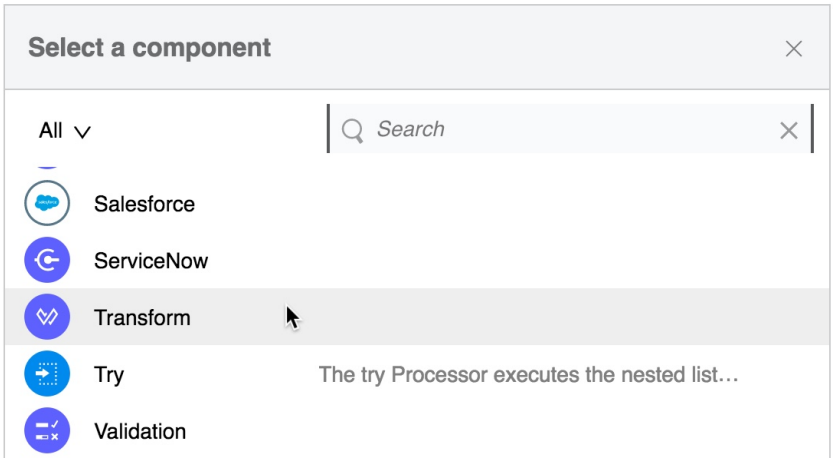
24. Return to flow designer and open the American Flights API card.
25. Click the Input tab and examine the Mule event data.
26. Click the Output tab; you should see payload data.



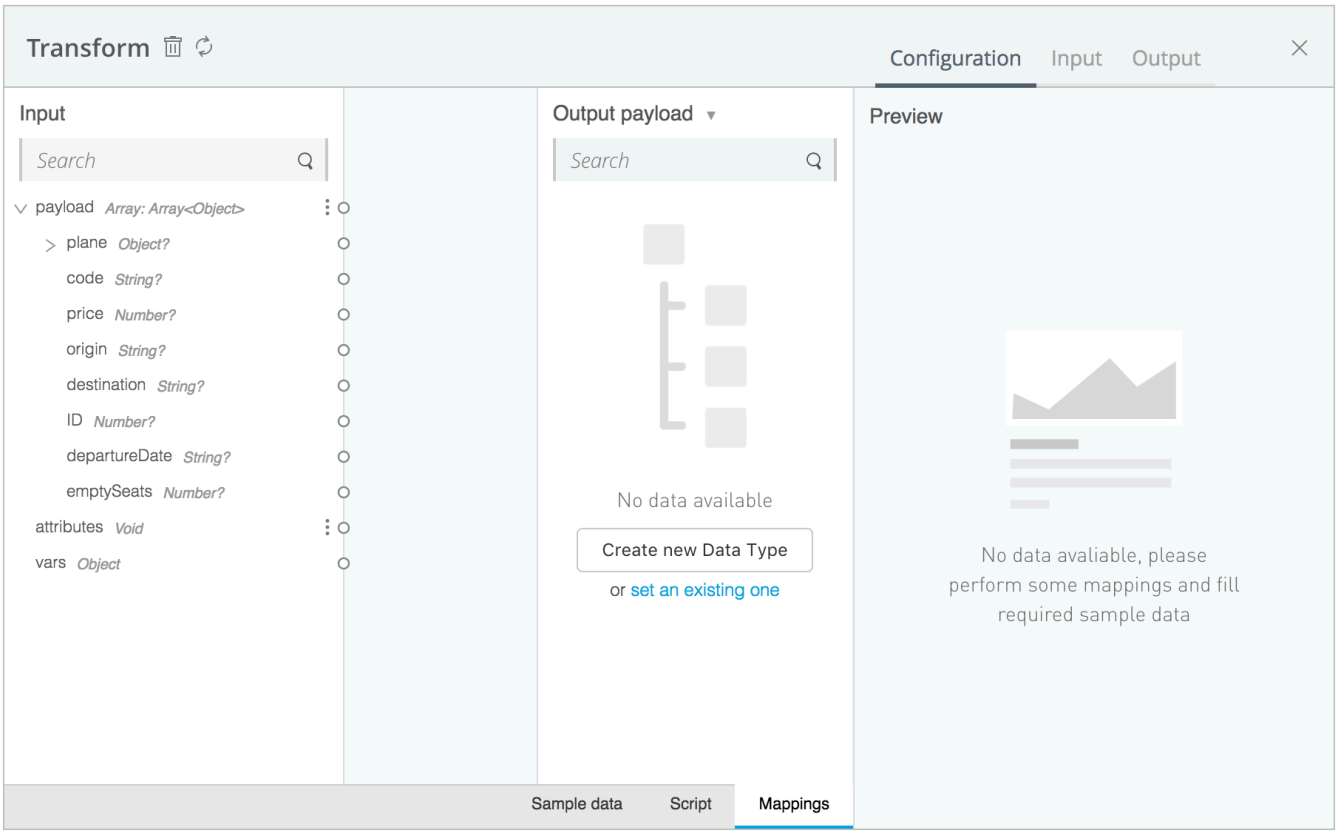
27. Close the card.

Add and configure a component to transform the data

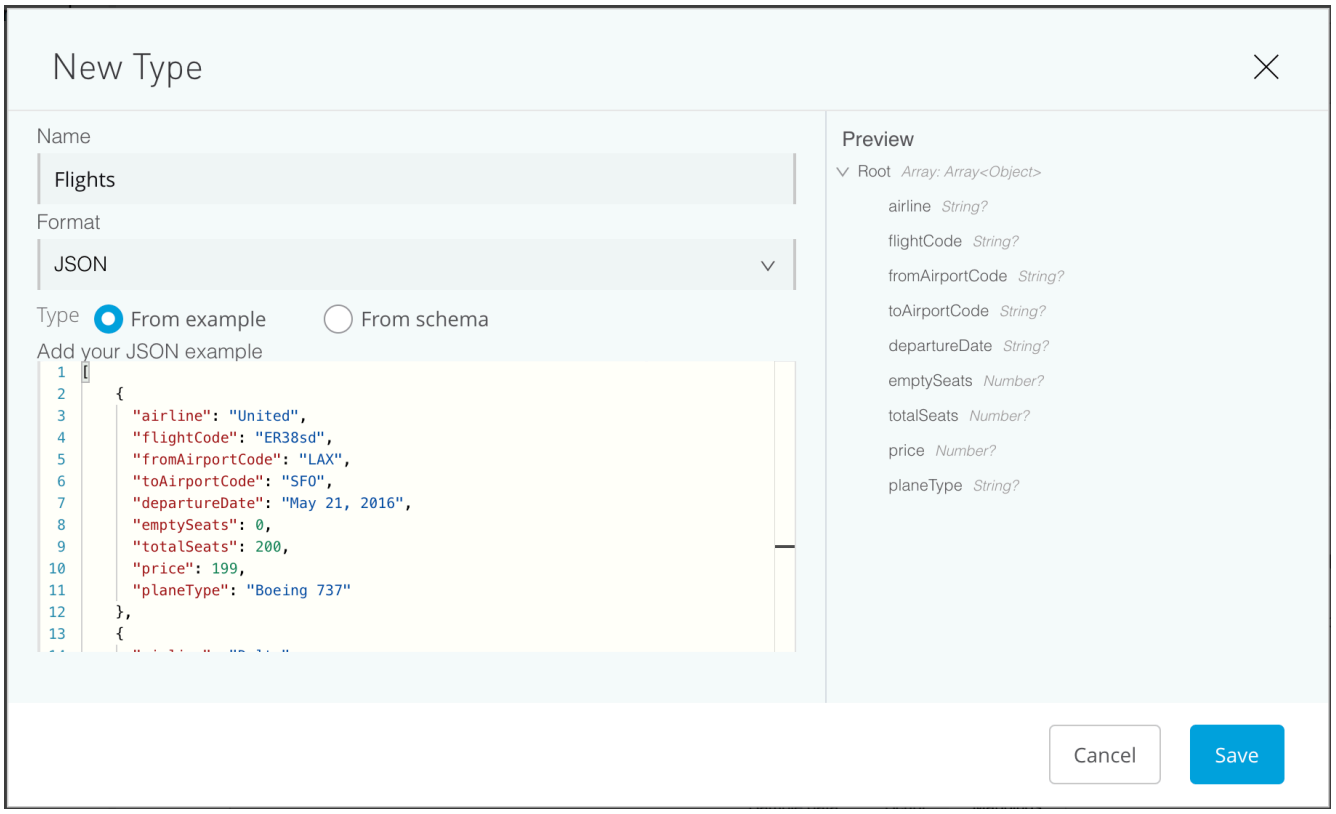
28. Click the add button in the flow.
29. In the Select a component dialog box, select Transform.



30. In the Transform card, look at the Mule event structure in the input section.
31. In the output section, click the Create new Data Type button.

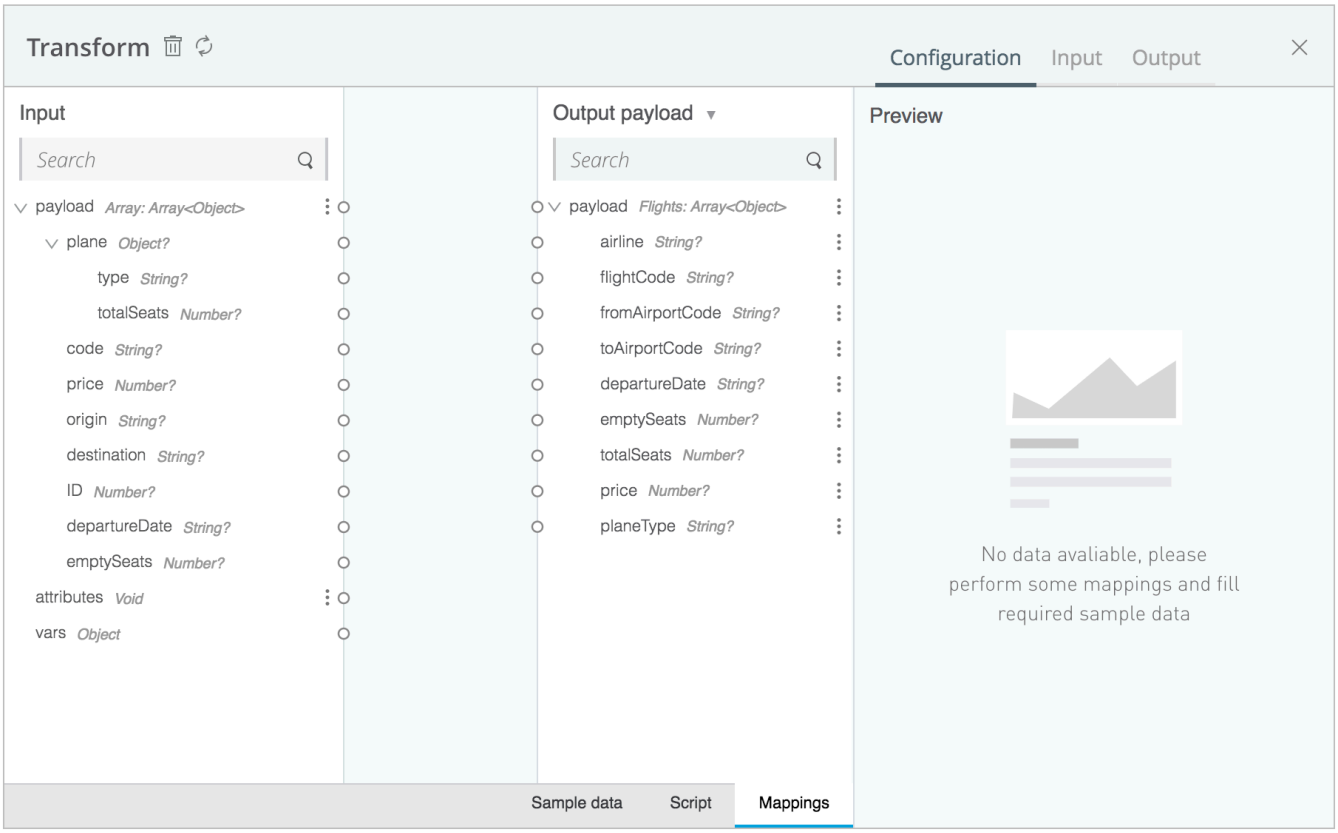


32. In the New Type dialog box, set the following values:
    - Name: Flights
    - Format: JSON
    - Type: From example
33. In the computer's file explorer, return to the student files folder and locate the flights-example.json file in the examples folder.
34. Open the file in a text editor and copy the code.
35. Return to flow designer and paste the code in the section to add your JSON example.



36. Click Save.

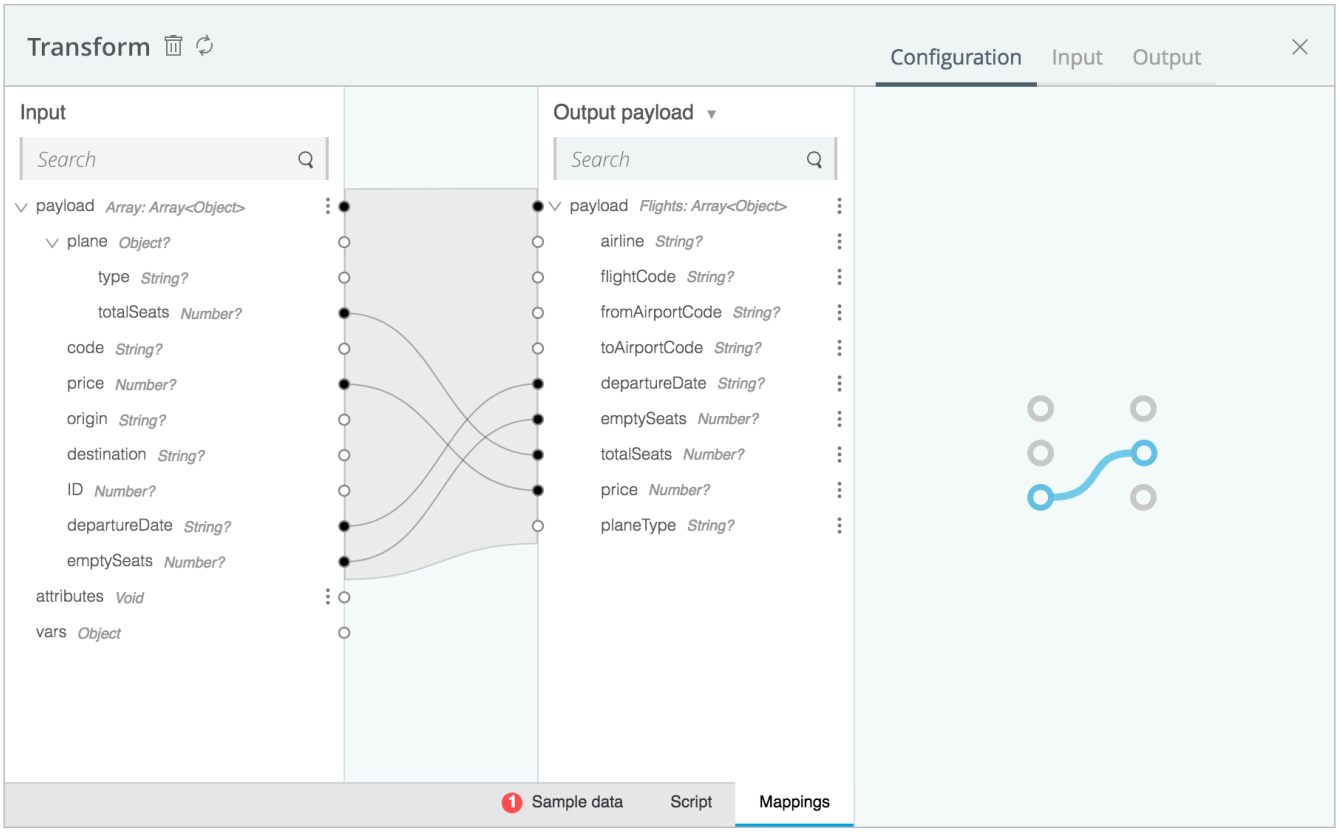
37. In the input section, expand the plane object.



Create the transformation

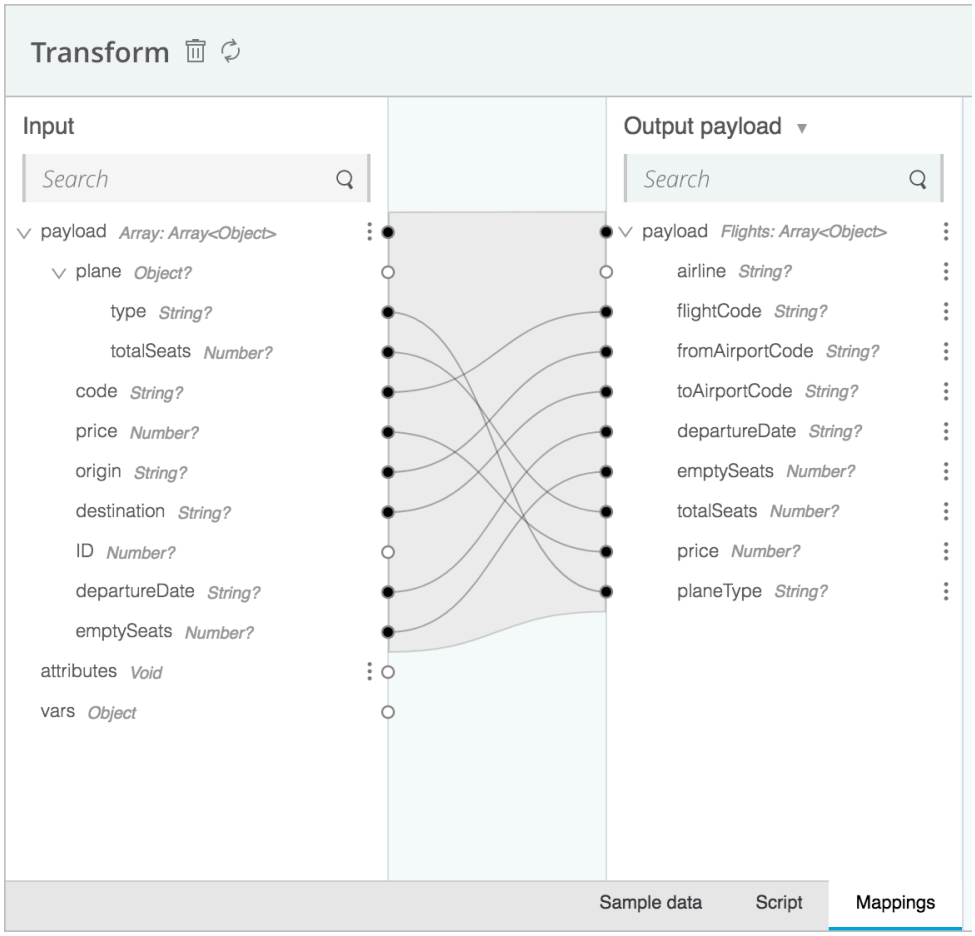
38. Map fields with the same names by dragging them from the input section and dropping them on the corresponding field in the output section.

    - price to price
    - departureDate to departureDate
    - plane > totalSeats to totalSeats
    - emptySeats to emptySeats

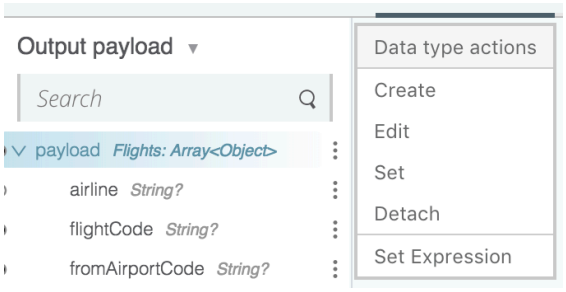


39. Map fields with different names by dragging them from the input section and dropping them on the corresponding field in the output section.

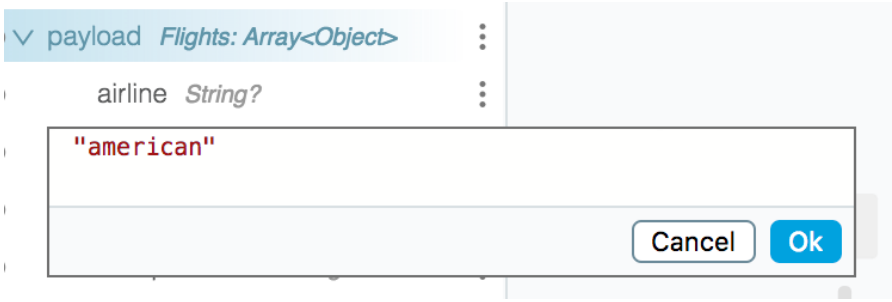
- plane > type to planeType
- code to flightCode
- origin to fromAirport
- destination to toAirport



40. In the output section, click the options menu for the airline field and select Set Expression.

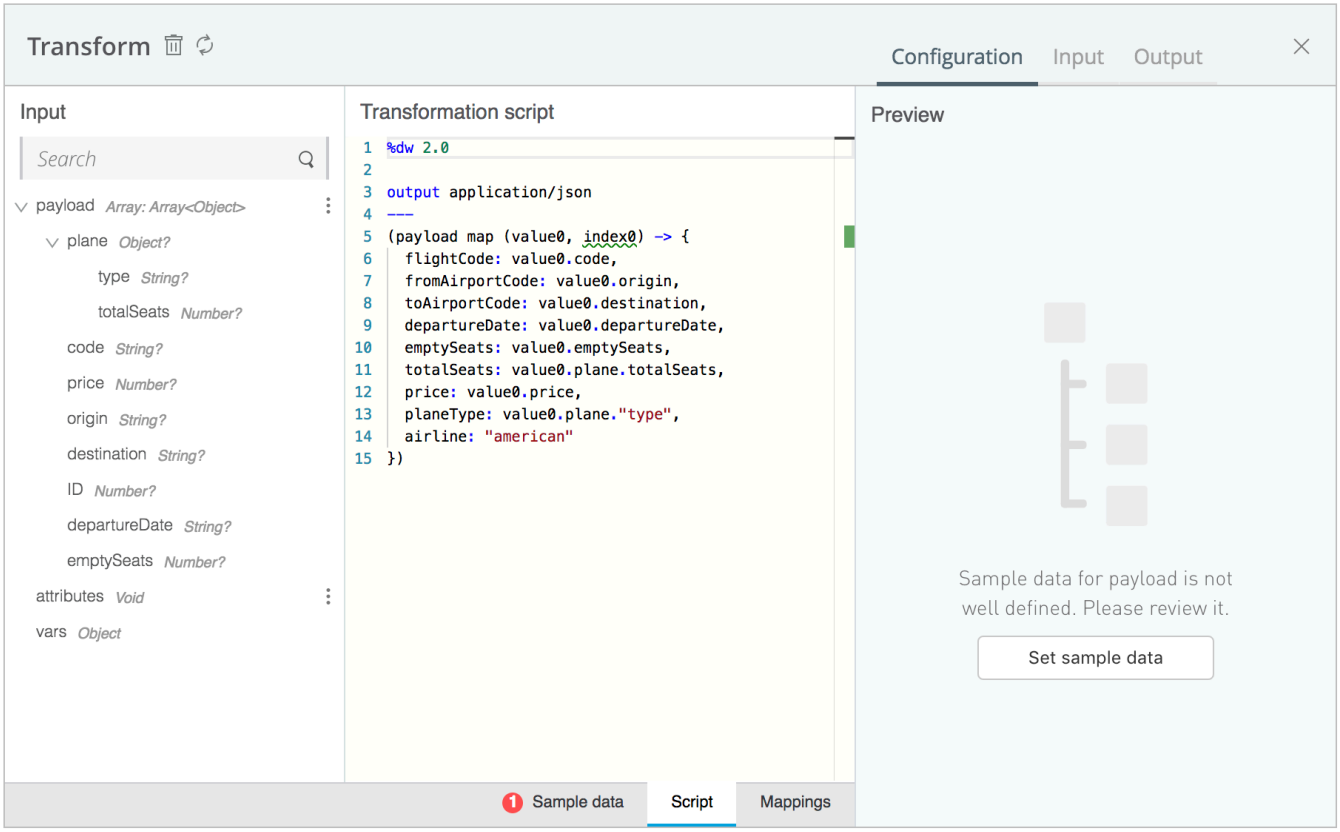


41. Change the value from null to "american" and click OK.



42. Click the Script tab at the bottom of the card; you should see the DataWeave expression for the transformation.

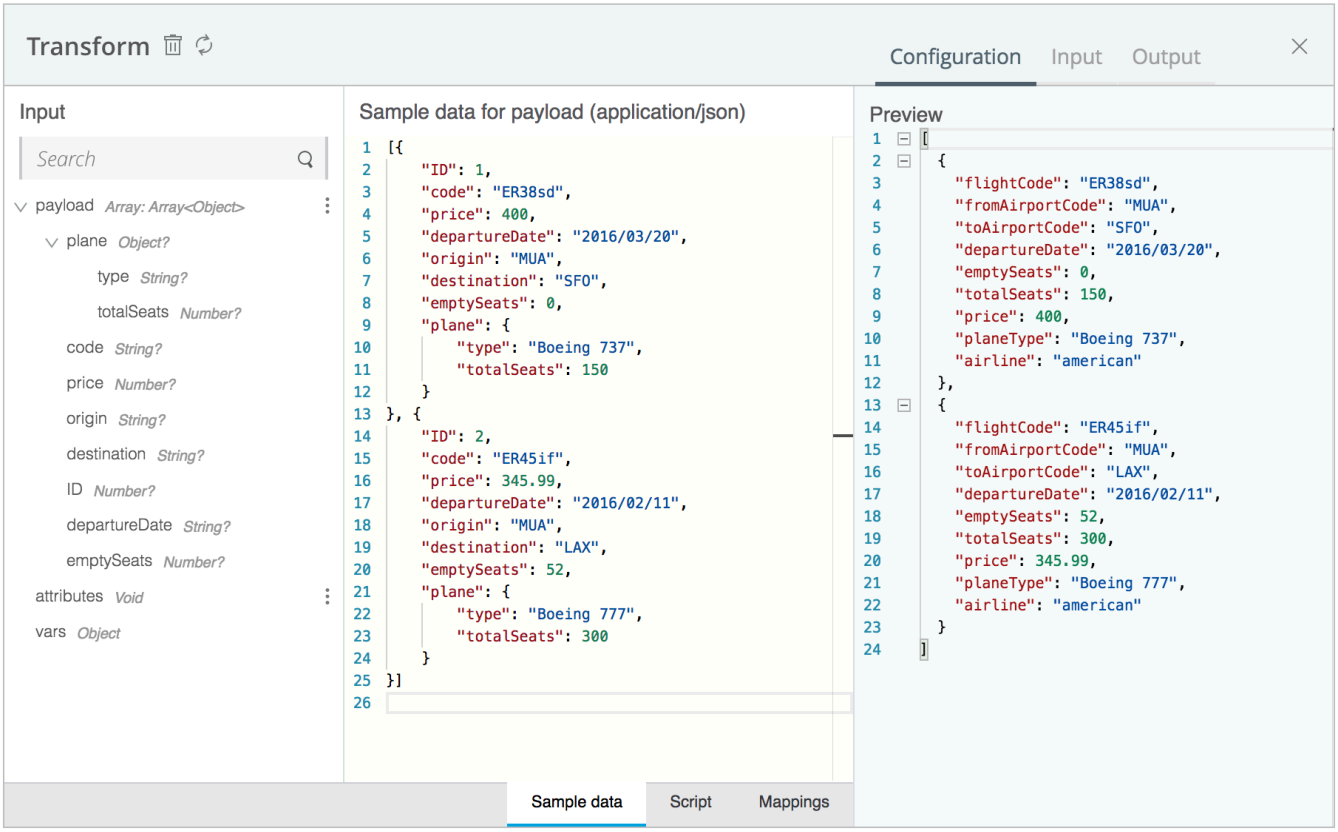
    Note: You learn to write DataWeave 1.0 expressions later in this course. You can also learn to write DataWeave expressions in the Flow Design course and the DataWeave course.



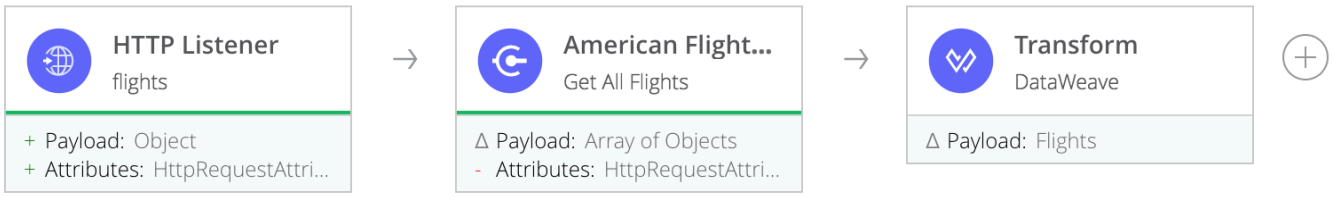
Add sample data

43. Click the Set sample data button in the preview section.
44. In the computer's file explorer, return to the student files folder and locate the american-flights-example.json file in the examples folder.

45. Open the file in a text editor and copy the code.
46. Return to flow designer and paste the code in the sample data for payload section.

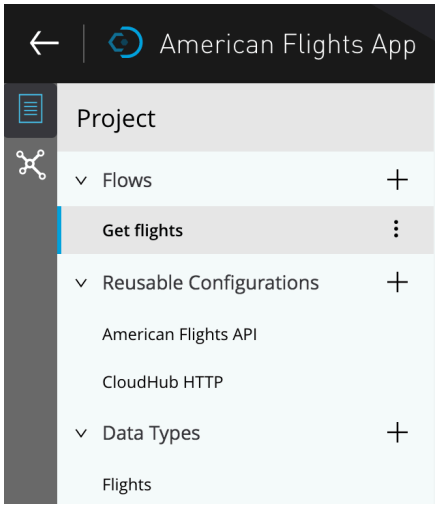


47. Look at the preview section, you should see a sample response for the transformation.
48. Close the card.



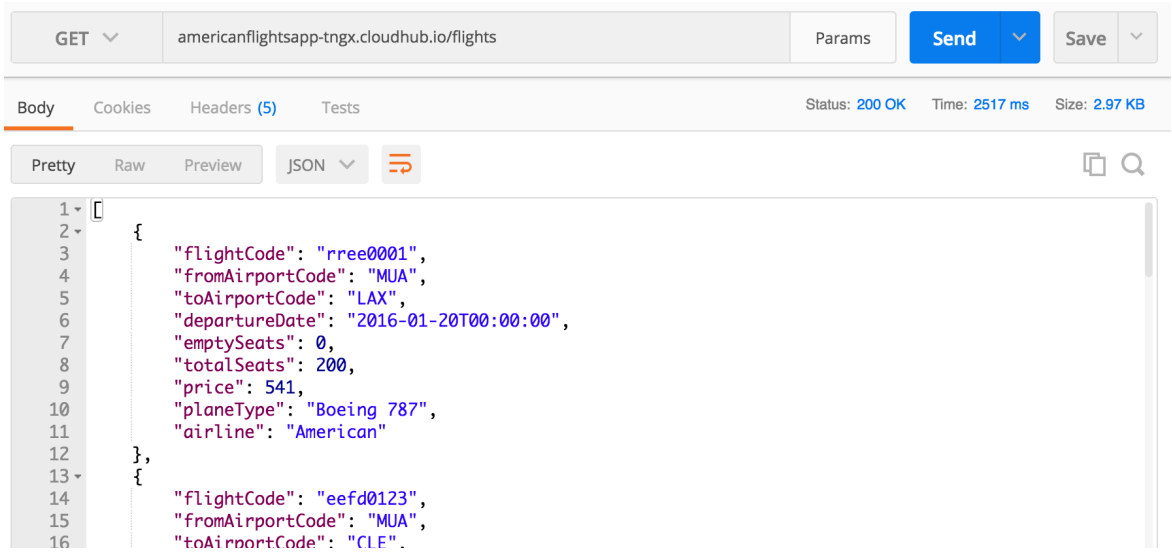
Locate the data type and configuration definitions

49. Locate the connector configurations and the new Flight data type in the project explorer.



Test the application

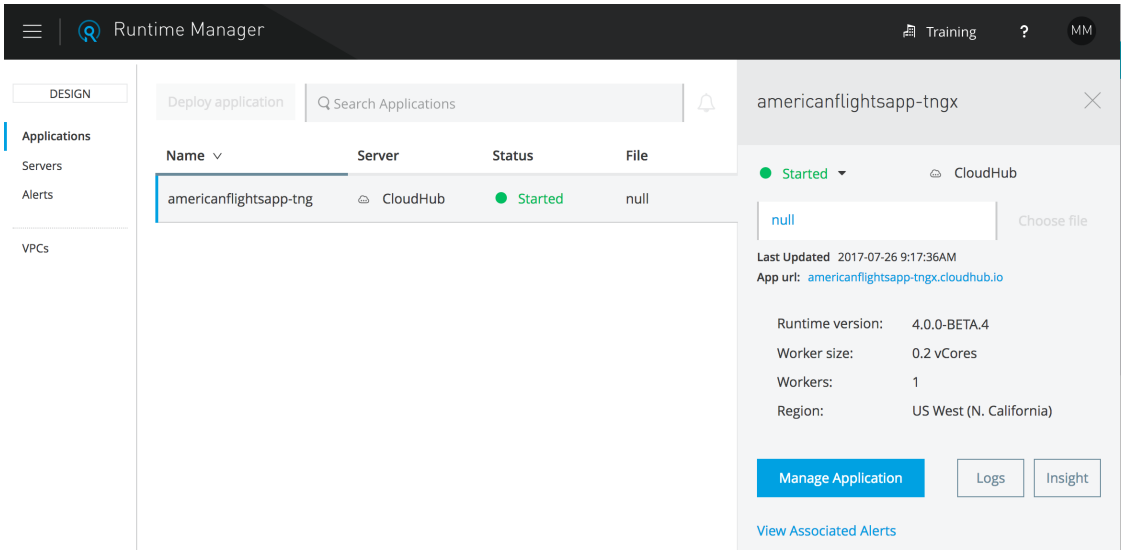
50. Deploy the project.
51. Return to Postman and click Send to make another request to http://americanflightsapp-xxxx.cloudhub.io/flights; you should see all the flight data as JSON again but now with a different structure.



Stop the application

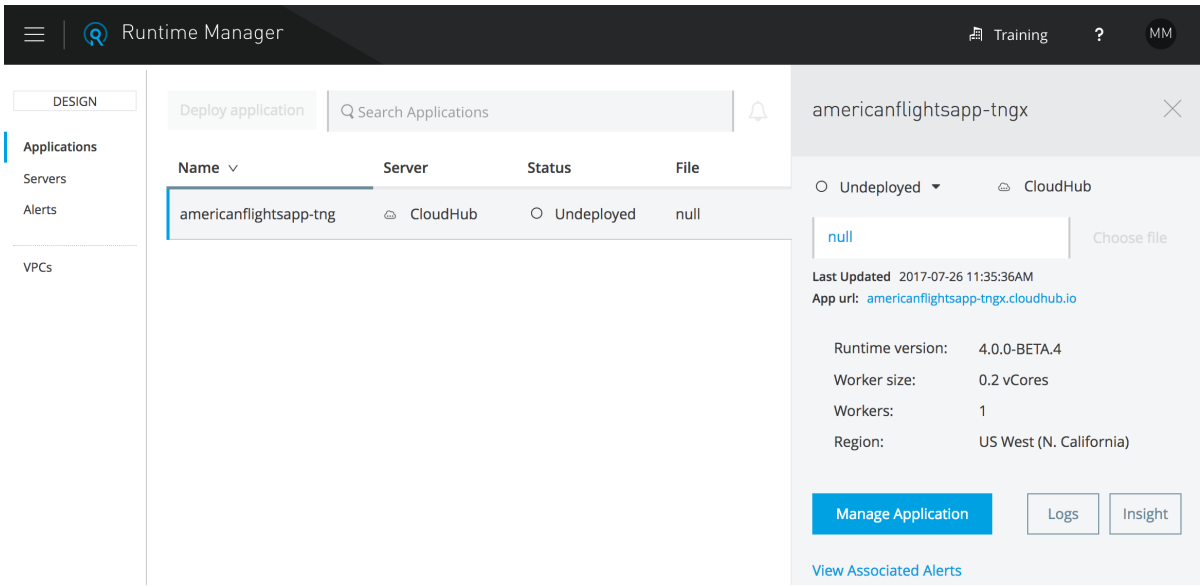
52. Return to Runtime Manager.
53. In the left-side navigation, click Applications.

54. Select the row with your application; you should see information about the application displayed on the right side of the window.

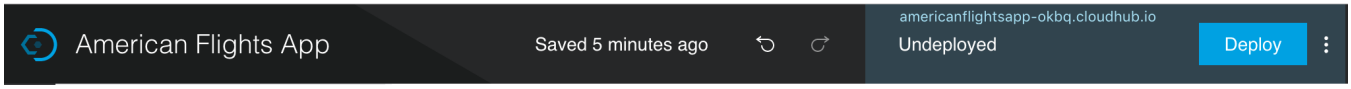


55. Click the drop-down menu button next to Started and select Stop; the status should change to Undeployed.

Note: You can deploy it again from flow designer when or if you work on the application again.



56. Close Runtime Manager.
57. Return to flow designer; you should see the application is undeployed.



58. Return to Design Center.