



MuleSoft®

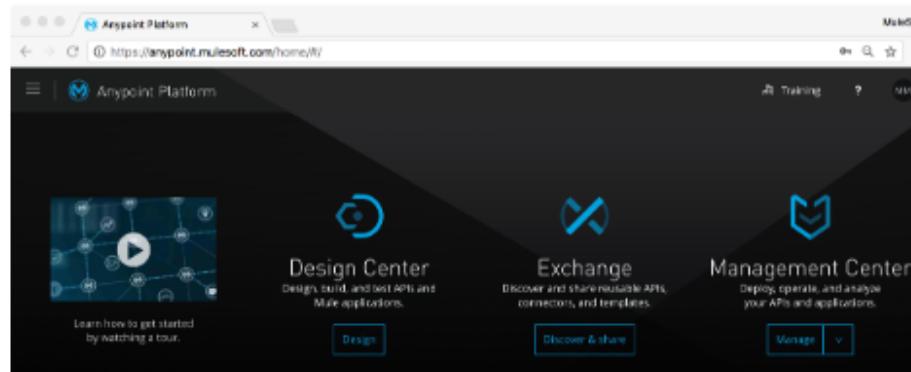
Module 2: Introducing Anypoint Platform



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



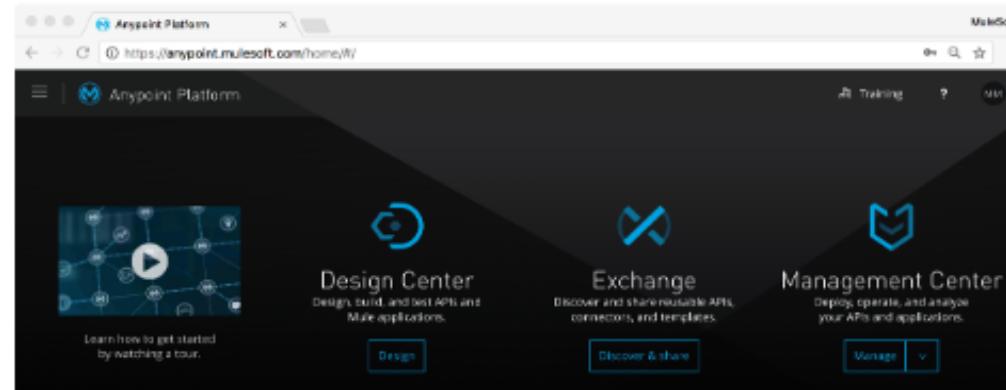
- Describe the benefits of Anypoint Platform and MuleSoft's approach to be successful with it
- Describe the role of each component in building application networks
- Navigate Anypoint Platform
- Locate APIs and other assets needed to build integrations and APIs in Anypoint Exchange
- Build basic integrations to connect systems using flow designer



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



- Describe the benefits of Anypoint Platform and MuleSoft's approach to be successful with it
- 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



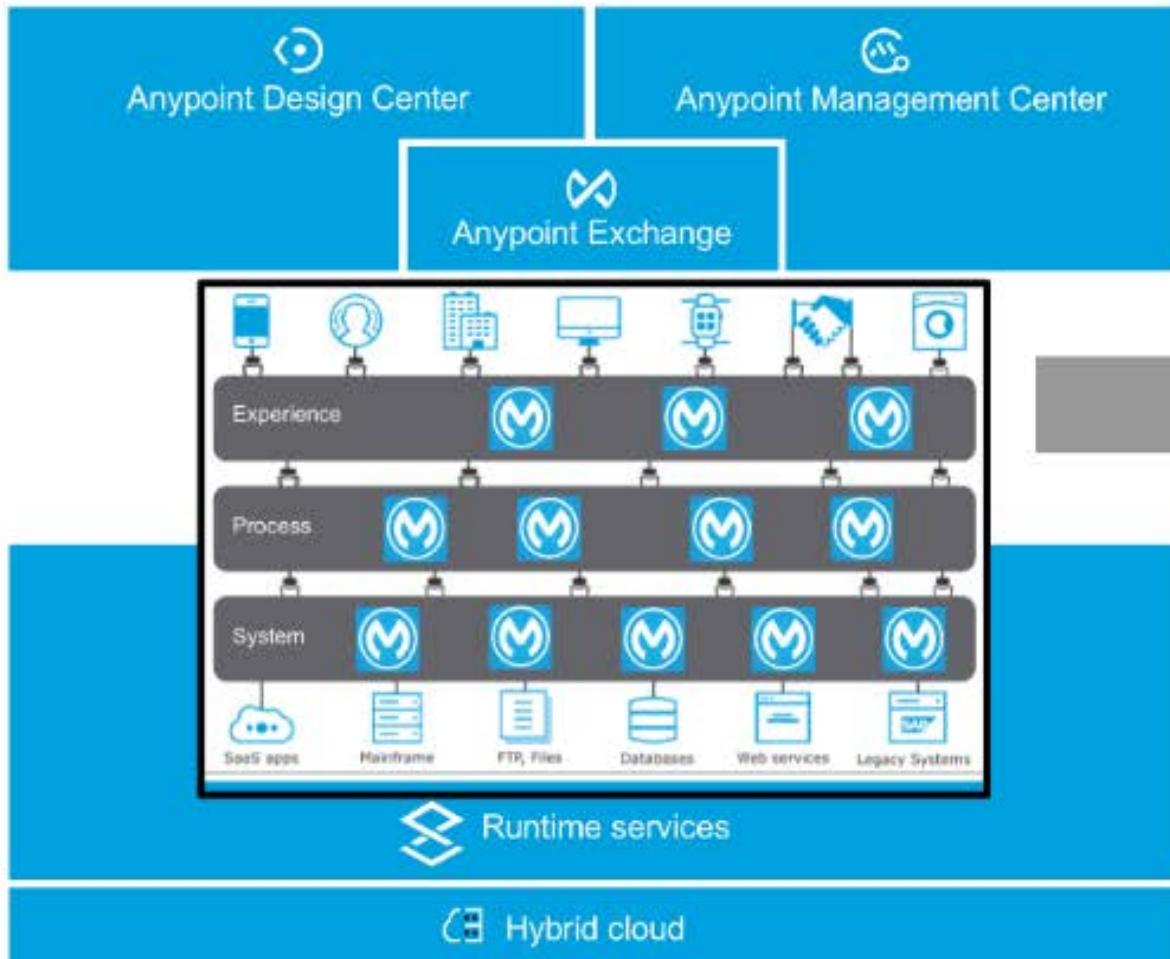
Introducing Anypoint Platform



Anypoint Platform uniquely enables the building of an application network

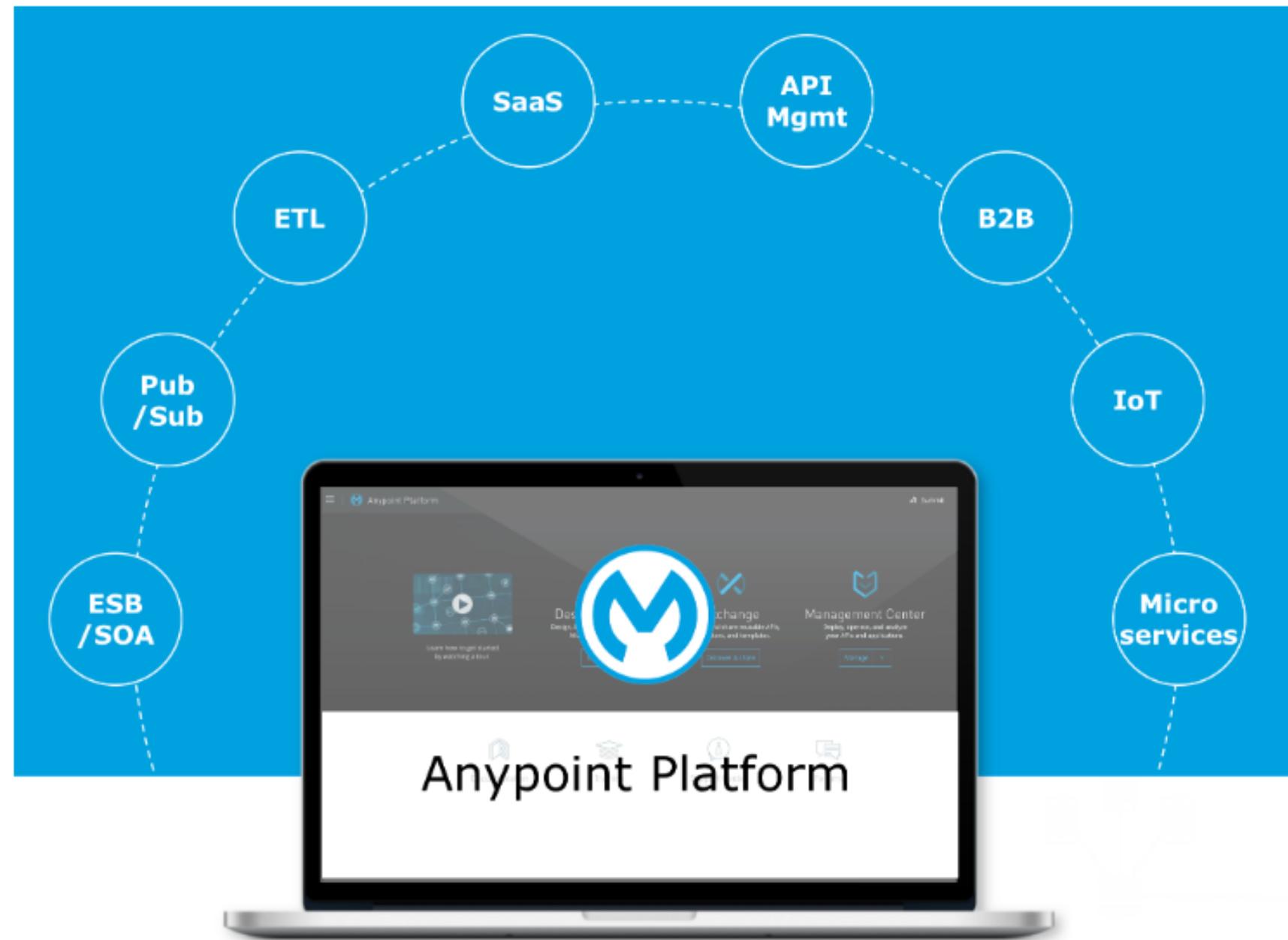


Anypoint Platform



Application network

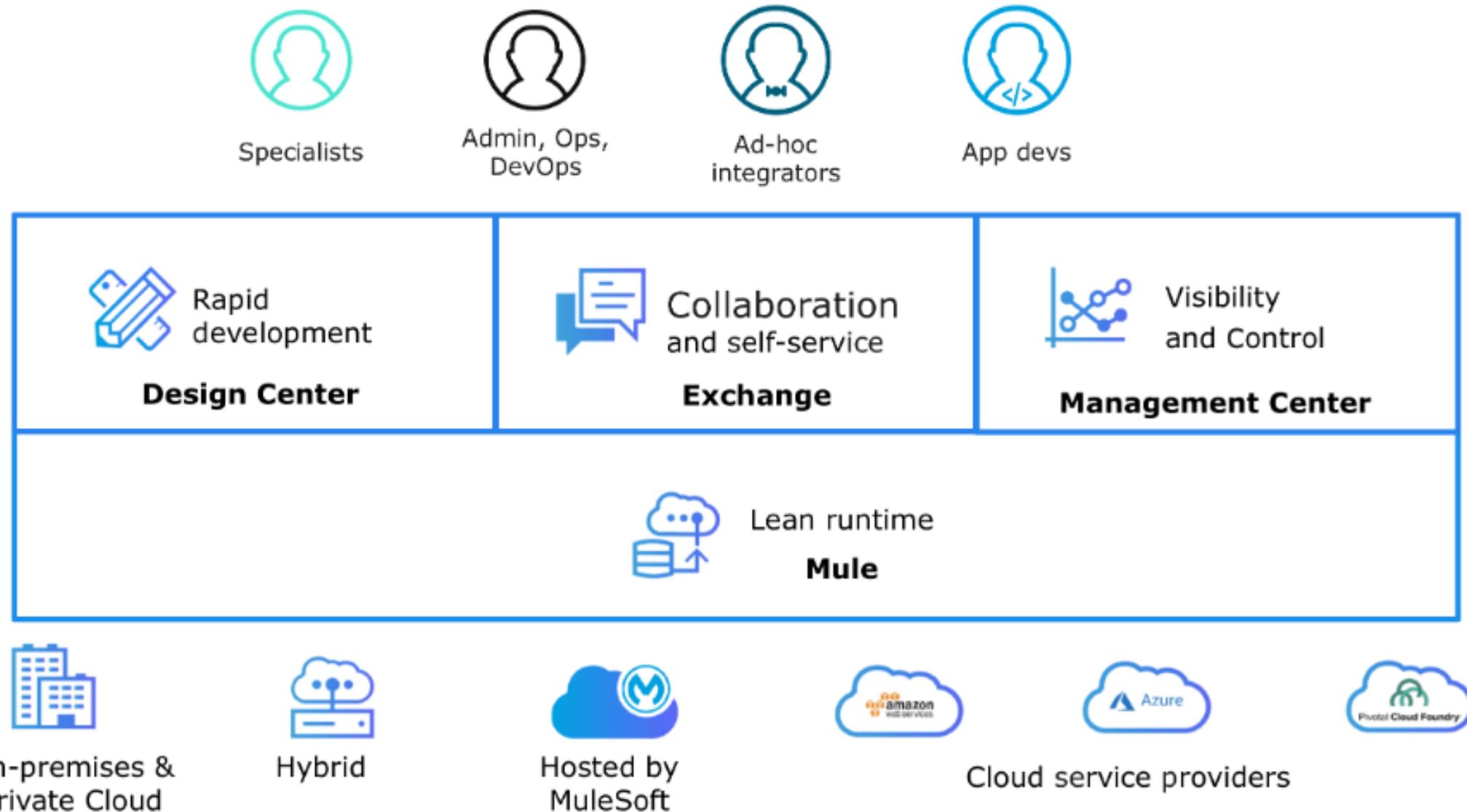




The most advanced enterprise platform for designing, developing and managing APIs and integrations

- Uniquely built as a single product
- Deploy anywhere
- Wide range of use cases

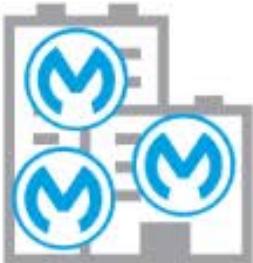
1 Design Center, 1 Mgmt Center, 1 Runtime





Anypoint Design Center

Anypoint Management Center



On-prem



Private cloud



fully managed iPaaS



Databases



FTP, Files



Web services



SaaS Apps



On-prem Apps



Social Apps

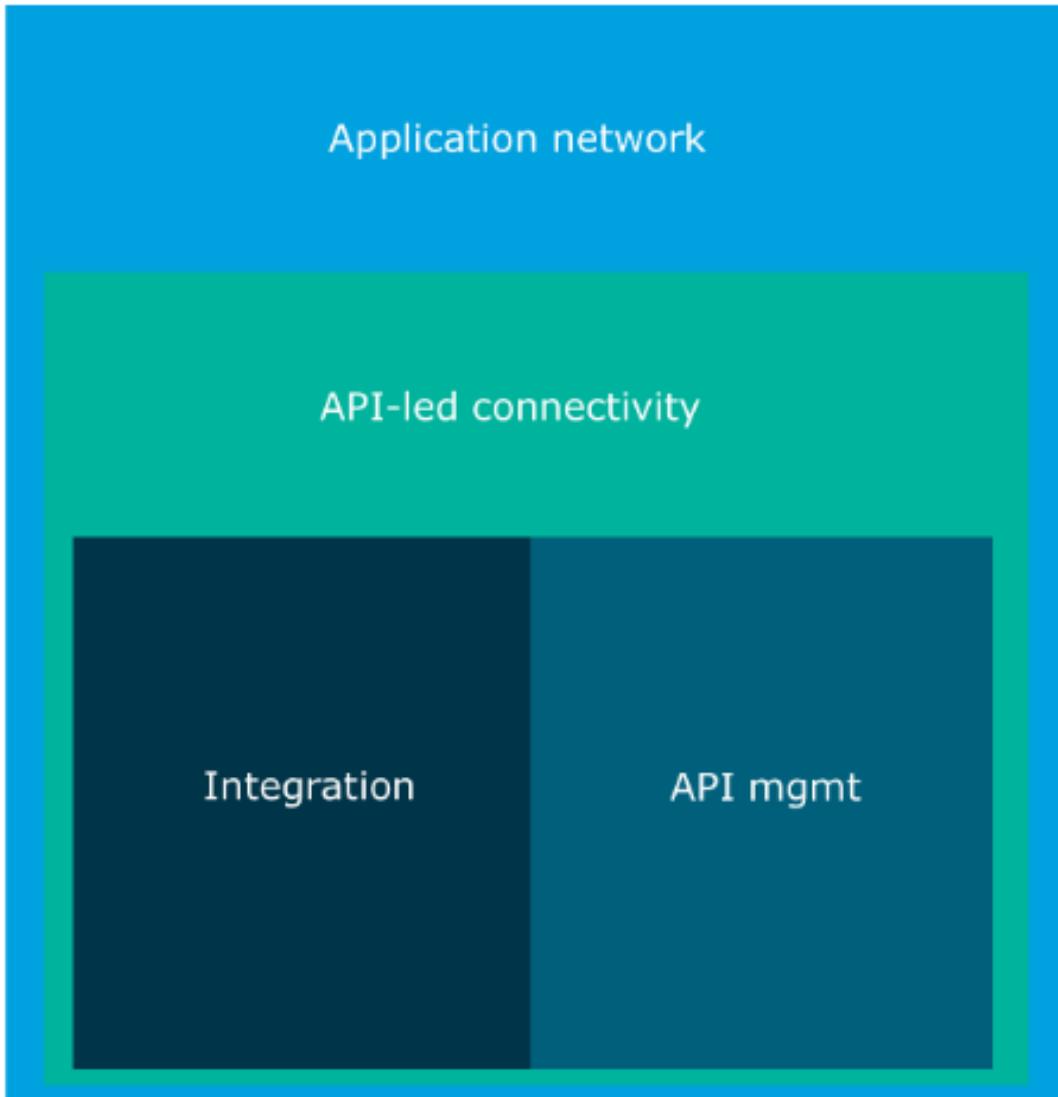


IoT



Partners

Unique benefits for each layer

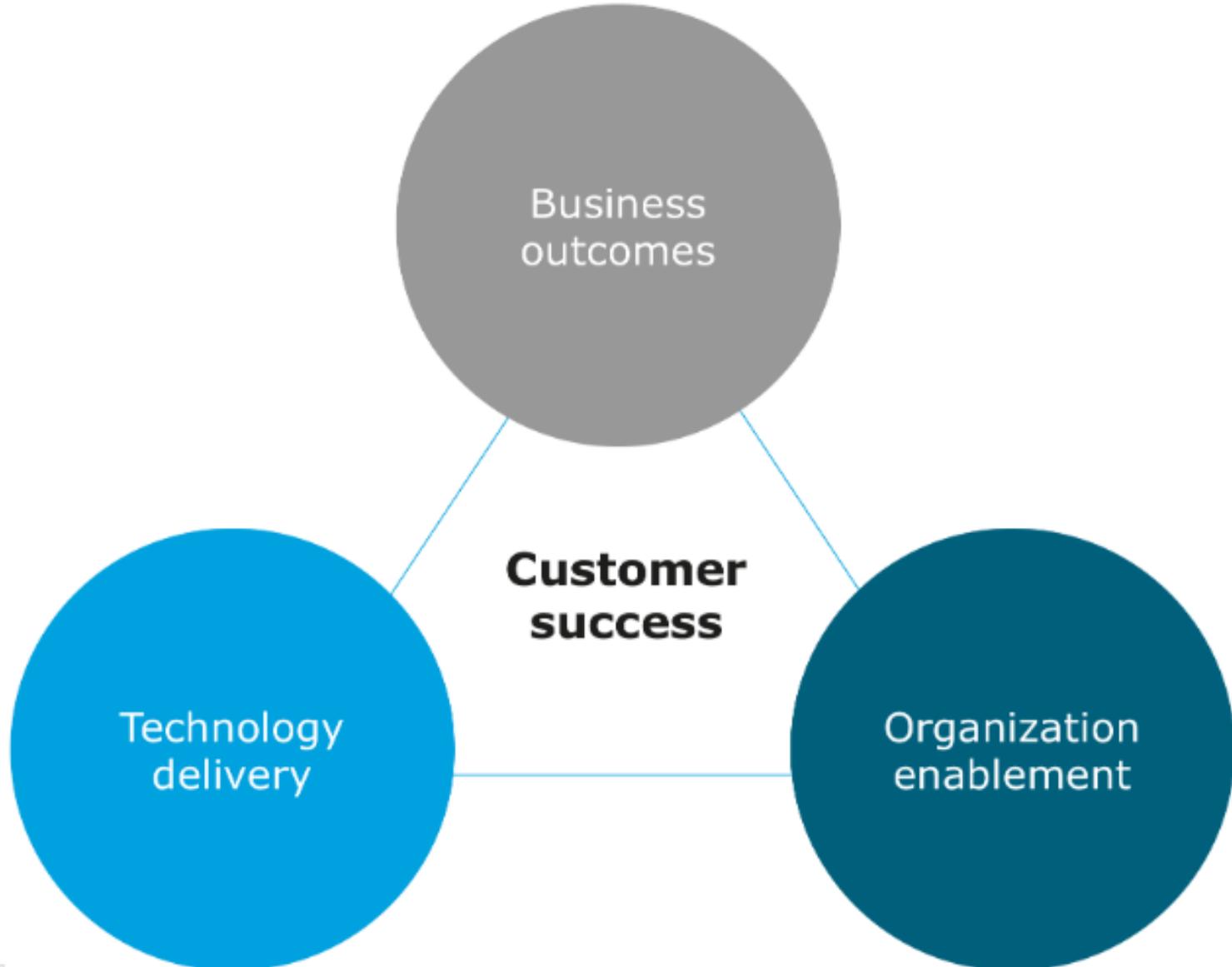


-  Speed of delivery
-  Actionable visibility
-  Secure by design
-  Future-proof architecture
-  Intentional self-service

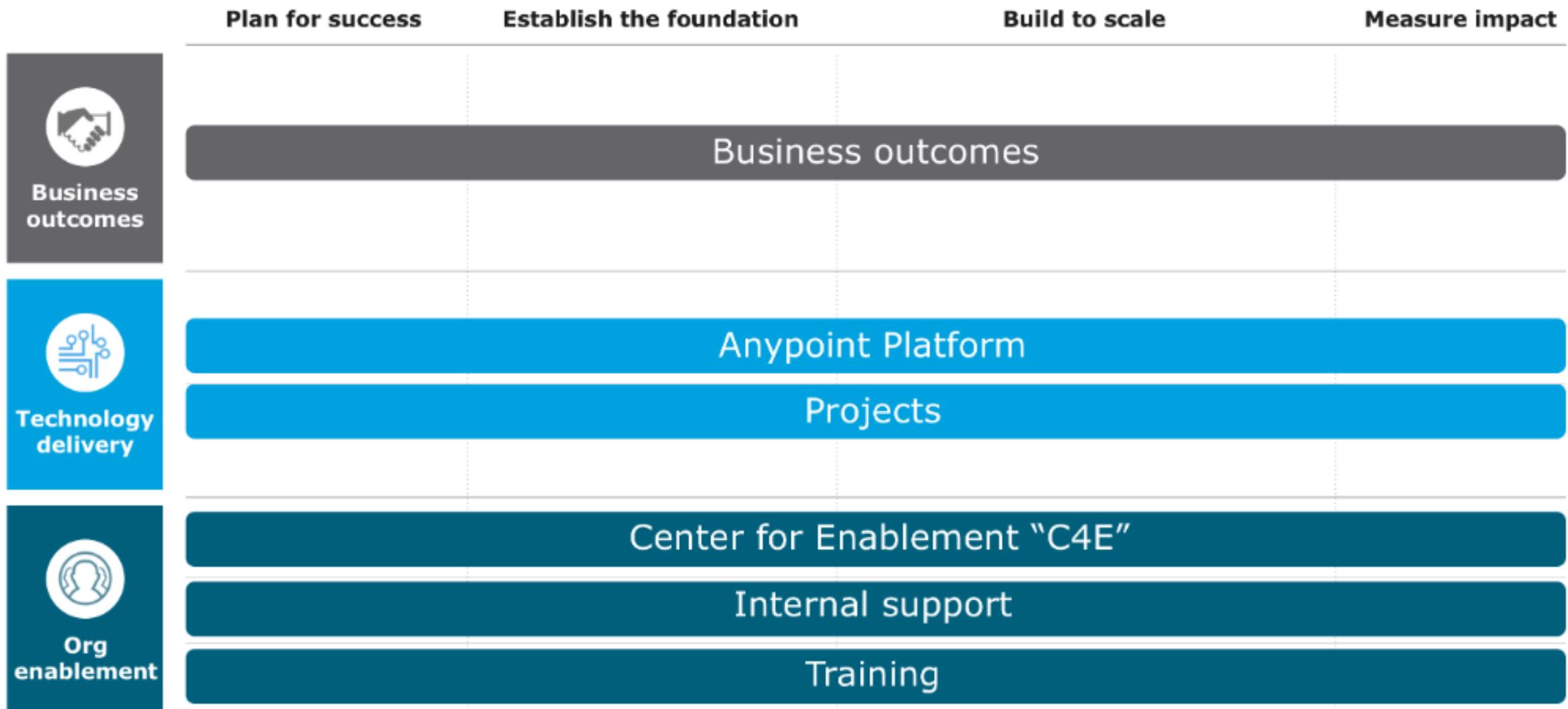
Achieving success with Anypoint Platform



MuleSoft's approach is centered around 3 core pillars



Defining the path to achieve success



MuleSoft has a blueprint for you to follow

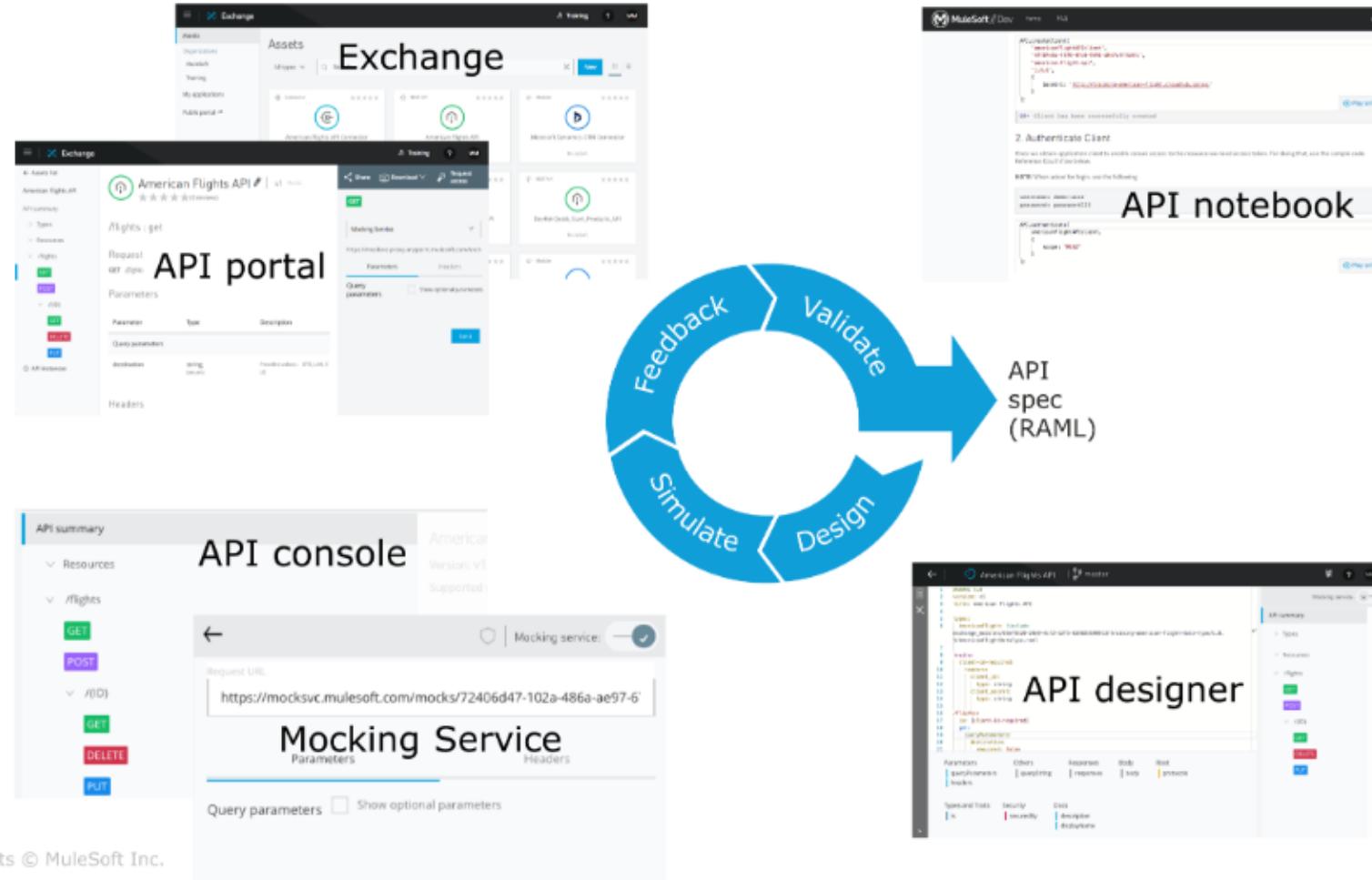


	Plan for success	Establish the foundation	Build to scale	Measure impact
 Business outcomes	Agree on business outcomes and KPIs Develop the overall success plan	Monitor and manage	Refresh the success plan	Measure business outcomes
 Technology delivery	Define Anypoint platform vision and roadmap Design Anypoint platform architecture and implementation plan	Deploy Anypoint Platform	Refine and scale Anypoint Platform	Measure Anypoint platform KPIs
	Prioritize IT projects and quick wins Staff and onboard the project teams	Define reference architecture Launch initial projects and quick wins	Onboard additional project teams Launch additional projects	Measure project KPIs
	Assess integration capabilities Establish the C4E operating model	Build and publish foundational assets Evangelize	Drive consumption	Measure C4E KPIs
 Org enablement	Onboard MuleSoft Determine the internal support operating model	Staff, train and launch team Publish support guidance and self-serve materials	Monitor Anypoint Platform	Measure support KPIs
	Agree on initial roles Train the initial team(s)	Develop the broader training plan Launch experiential learning opportunities	Update training plan	Conduct skills assessment

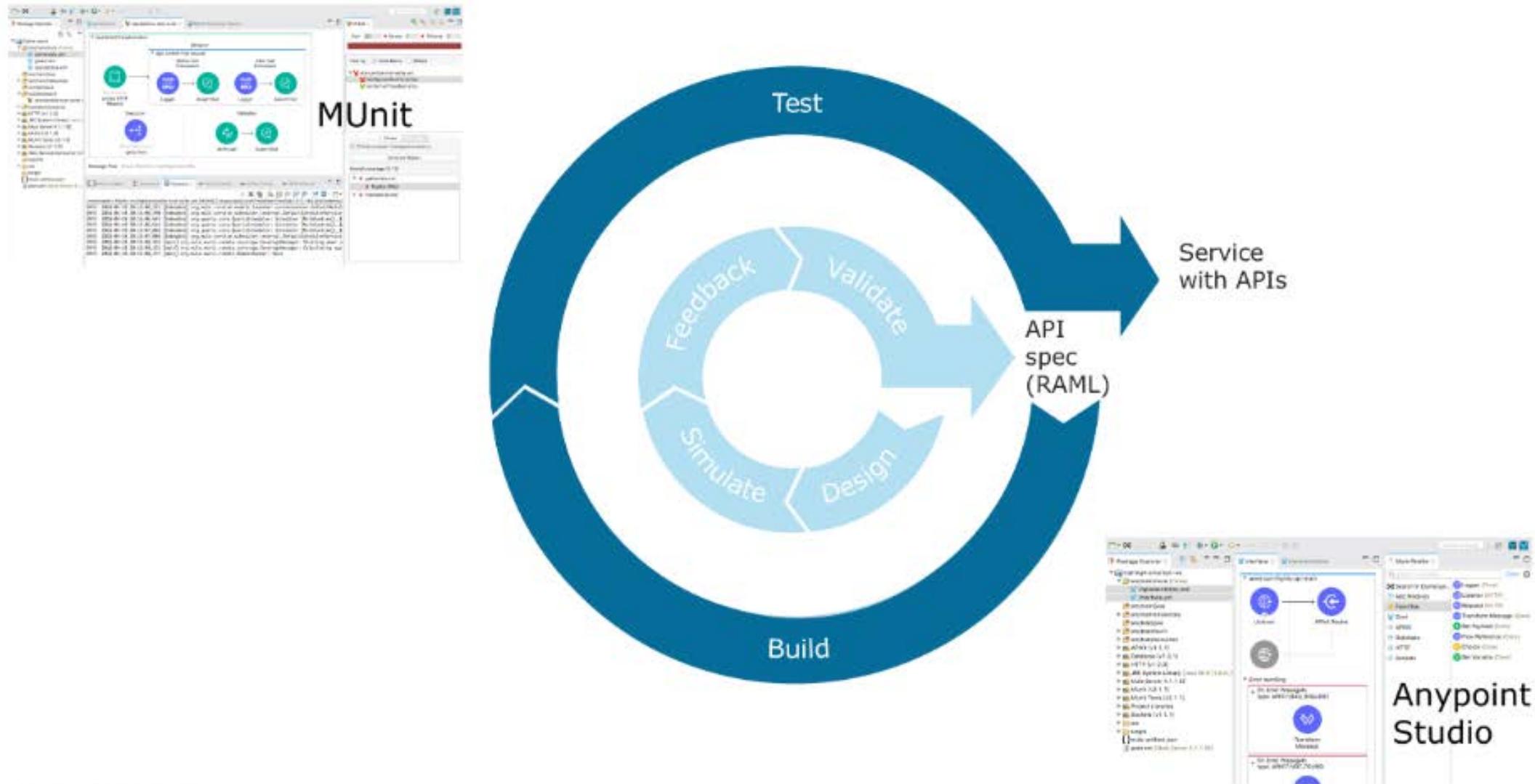
Introducing the components of Anypoint Platform



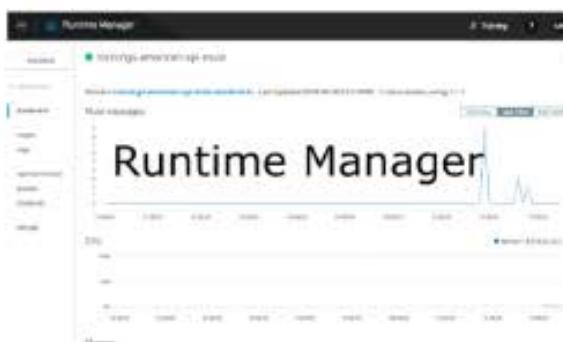
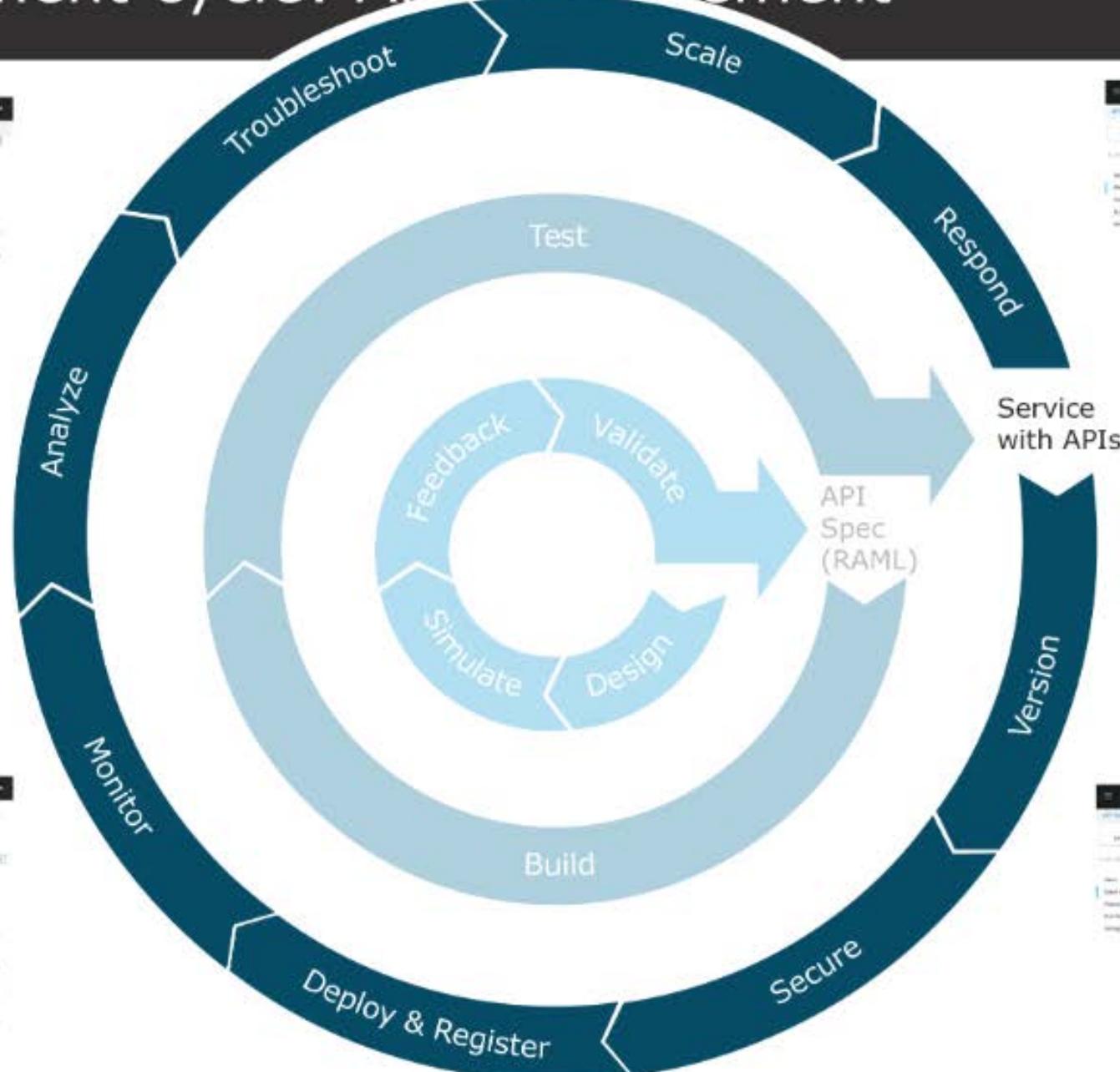
API development cycle: API specification



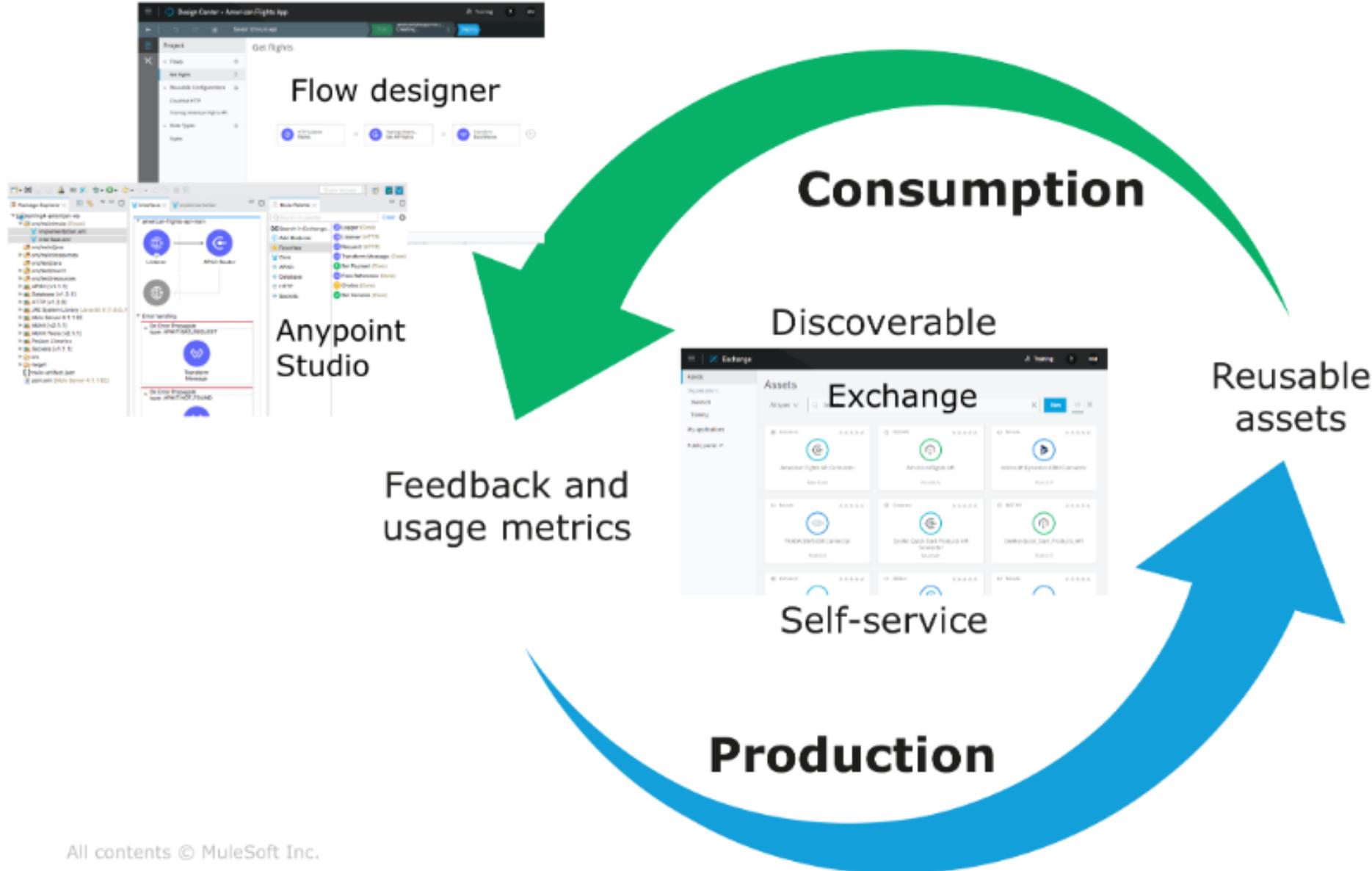
API development cycle: API implementation



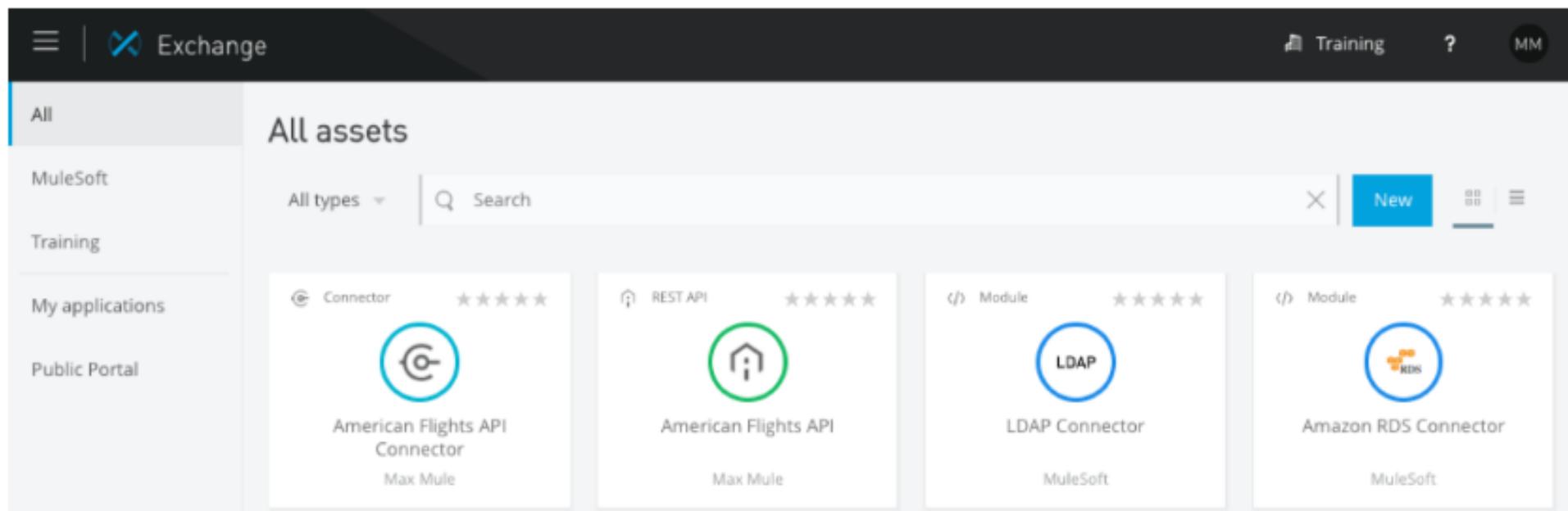
API development cycle: API management



API lifecycle: Discovery and consumption



- A library of assets
- The central repository that is critical to the success of building an application network
- Ensures assets are published somewhere they can be discovered and reused



The screenshot shows the Anypoint Exchange web interface. The top navigation bar includes a menu icon, the title "Exchange", a "Training" link, a help icon, and a user profile icon. The left sidebar has a "All" tab selected, followed by "MuleSoft", "Training", "My applications", and "Public Portal". The main content area is titled "All assets" and features a search bar with "All types" dropdown and "Search" input, along with a "New" button and filter icons. Below the search bar, there are four asset cards: "American Flights API Connector" (Connector, 5 stars, Max Mule), "American Flights API" (REST API, 5 stars, Max Mule), "LDAP Connector" (Module, 5 stars, MuleSoft), and "Amazon RDS Connector" (Module, 5 stars, MuleSoft).

Type	Name	Rating	Provider
Connector	American Flights API Connector	★★★★★	Max Mule
REST API	American Flights API	★★★★★	Max Mule
Module	LDAP Connector	★★★★★	MuleSoft
Module	Amazon RDS Connector	★★★★★	MuleSoft

What does (and should) Exchange contain?



- MuleSoft-provided **public** assets available in all accounts to all users
 - You can work with MuleSoft to get APIs and connectors certified and added
- **Private** content only available to people in your org
 - Assets added by anyone in your org are added to your private Exchange
- Your organization should populate it to contain everything you need to build your integration projects
 - Including APIs, connectors, diagrams, videos, links, and more

All types
Connectors
Templates
Examples
REST APIs
SOAP APIs
HTTP APIs
RAML fragments
Custom

- When a REST API is added to Exchange, an **API portal** is automatically created for it
- An API portal has
 - Auto-generated **API documentation**
 - An **API console** for consuming and testing APIs
 - An **automatically generated API endpoint** that uses a **mocking service** to allow the API to be tested without having to implement it
- API portals can be shared with both internal and external users
- In the last module, you used a public API portal created from Anypoint Exchange for a private organization (Muletraining)

REST connectors in Anypoint Exchange

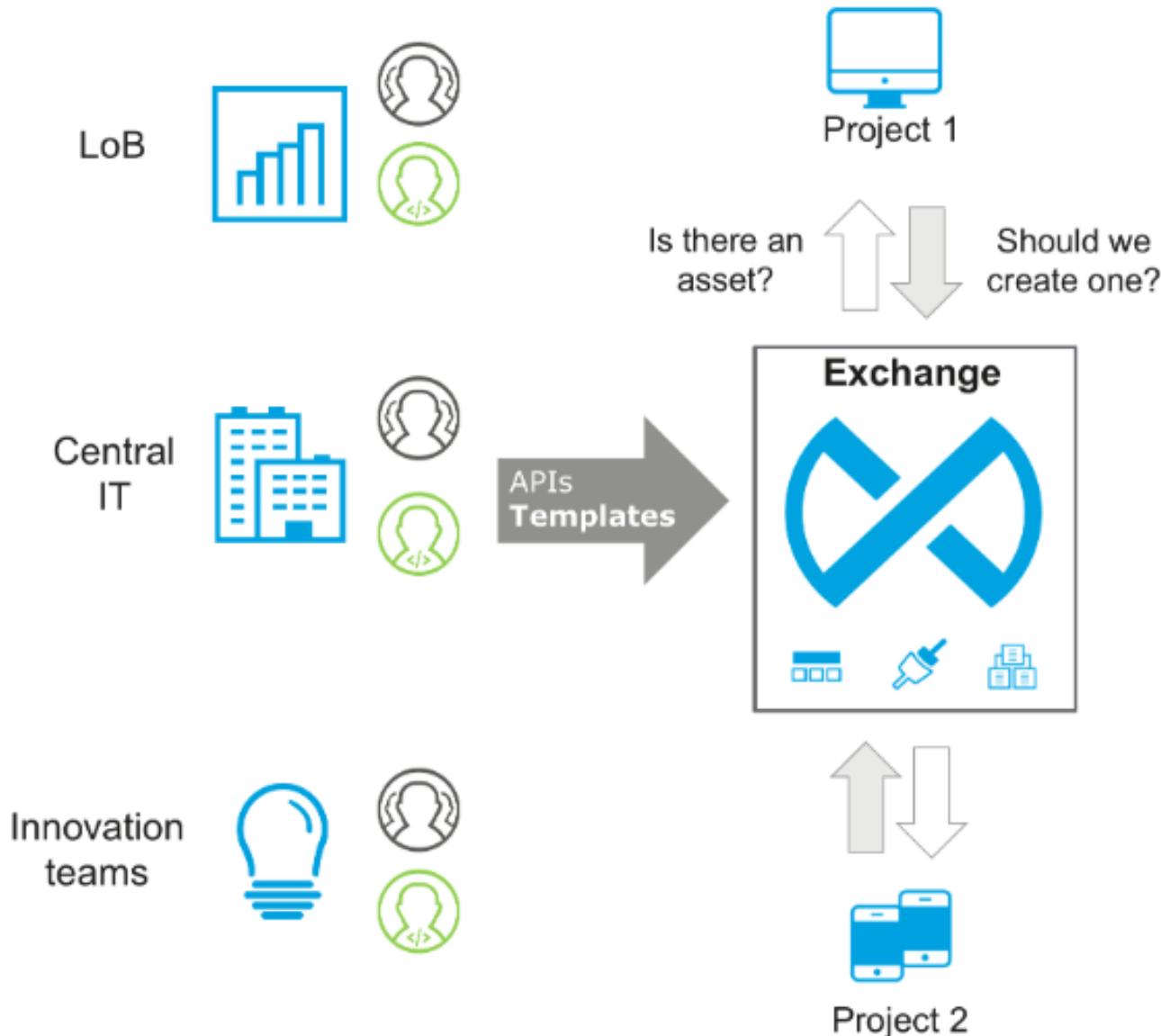


- When a RAML 1.0 API specification is added to Exchange, a **connector** is automatically created for it
 - The connector can be used in Mule applications to make calls to that API
 - REST Connect is the name of the technology that performs this conversion

A screenshot of the Anypoint Exchange web interface. The top navigation bar includes 'Training' and a search bar. The left sidebar has categories like 'All', 'MuleSoft', 'Training', 'My applications', and 'Public Portal'. The main area is titled 'All assets' and shows four items:

- American Flights API Connector (Connector, 5 stars, Max Mule)
- American Flights API (REST API, 5 stars, Max Mule)
- LDAP Connector (Module, 5 stars, MuleSoft)
- Amazon RDS Connector (Module, 5 stars, MuleSoft)

Using Exchange: Success of C4E in action



Walkthrough 2-1: Explore Anypoint Platform and Anypoint Exchange



- 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 in the public Exchange

A screenshot of the Anypoint Exchange web interface. The top navigation bar includes a menu icon, the 'Exchange' logo, a search icon, the word 'Training', a help icon, and a user icon. The left sidebar has a 'Assets' tab selected, followed by 'Organizations', 'MuleSoft', 'Training', 'My applications', and 'Public portal'. A search bar at the top right contains the text 'american'. Below the search bar, it says 'Showing results for "american". [Save this search](#)'. Three API assets are listed:

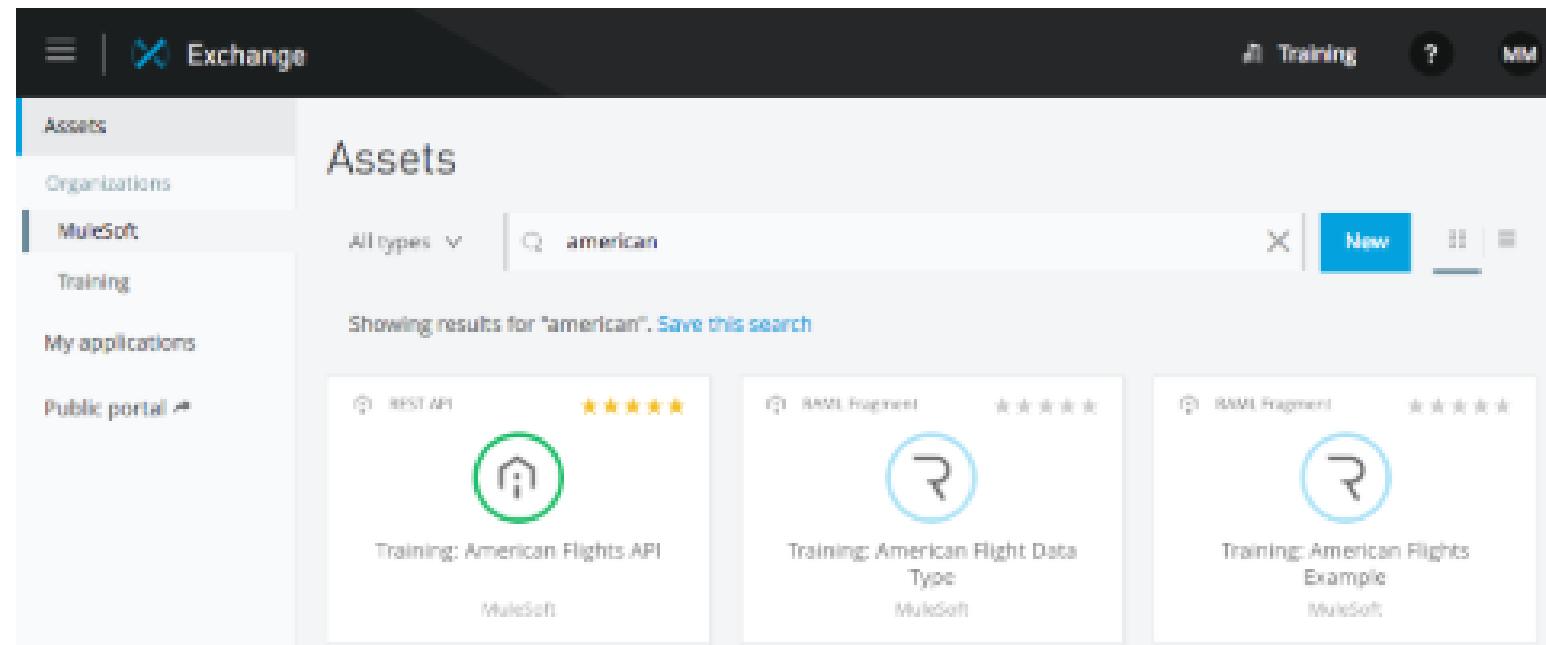
- REST API: Training: American Flights API (5 stars)
- RAML Fragment: Training: American Flight Data Type (5 stars)
- RAML Fragment: Training: American Flights Example (5 stars)

All contents © MuleSoft Inc.

Walkthrough 2-1: Explore Anypoint Platform and Anypoint Exchange

In this walkthrough, you get familiar Anypoint Platform. 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 in the public Exchange.



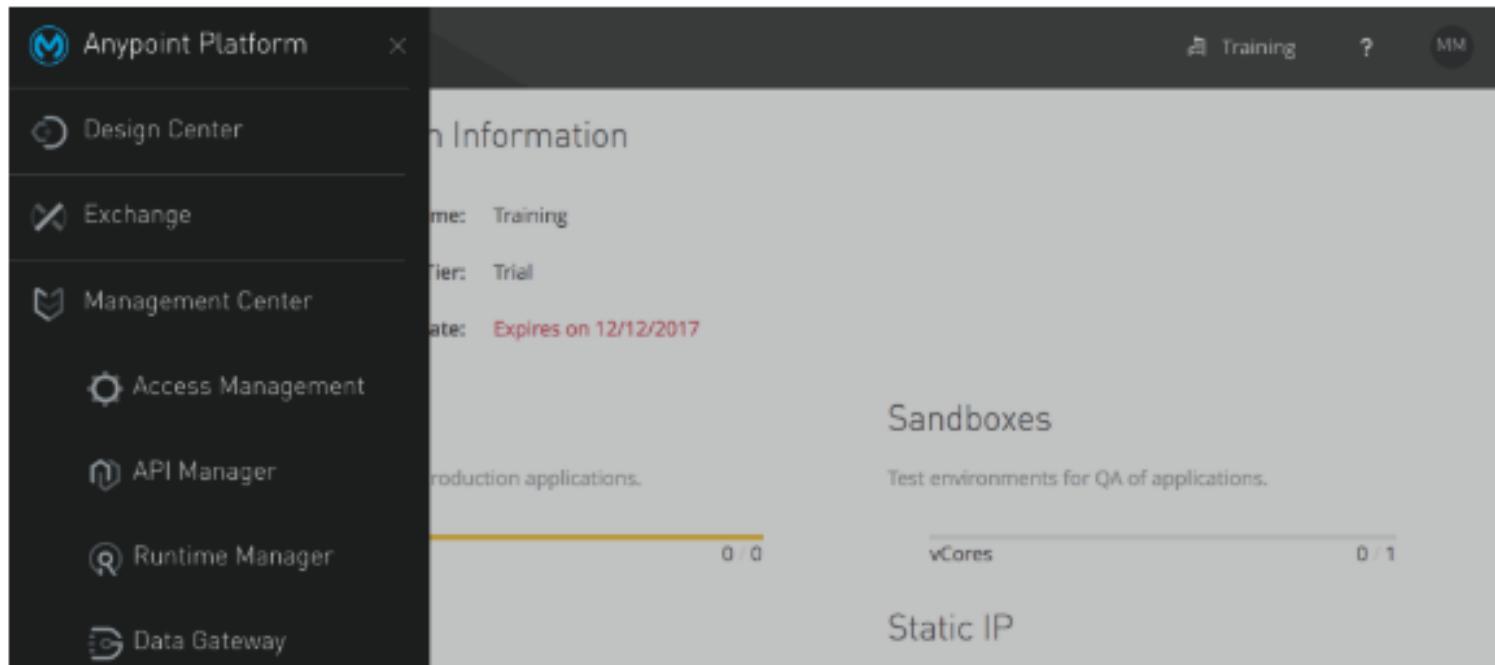
Return to Anypoint Platform

1. Return to Anypoint Platform at <https://anypoint.mulesoft.com> (not the public API portal you used last module!) 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 left-side navigation, select Users.
6. In the left-side navigation, select Environments.

The screenshot shows the 'Access Management' interface. On the left, a sidebar lists 'ACCESS MANAGEMENT' (Organization, Users, Roles, Environments), 'SETTINGS' (Runtime Manager), and 'SUBSCRIPTION' (Runtime Manager). The 'Environments' section is selected. The main area displays a table titled 'Environments' with two rows: 'Design' (Type: Design) and 'Sandbox' (Type: Sandbox). A blue button labeled 'Add environment' is visible. The top navigation bar includes 'Training', a help icon, and a 'MM' icon.

7. In the main menu, select Design Center.
8. Click the Create button and look at the options in the drop-down menu.

The screenshot shows the 'Design Center' interface. At the top, there's a search bar and a 'Projects' section. Below it is a table with columns 'Name', 'Project Type', and 'Last Update'. A 'Create' button is highlighted with a blue box. A dropdown menu from this button lists 'Mule Application', 'API specification', 'API fragment', and 'Get Anypoint Studio'. To the right of the dropdown is a small graphic of a chart.

9. In the main menu, select Runtime Manager.

10. If you get a Choose environment page, select Design.

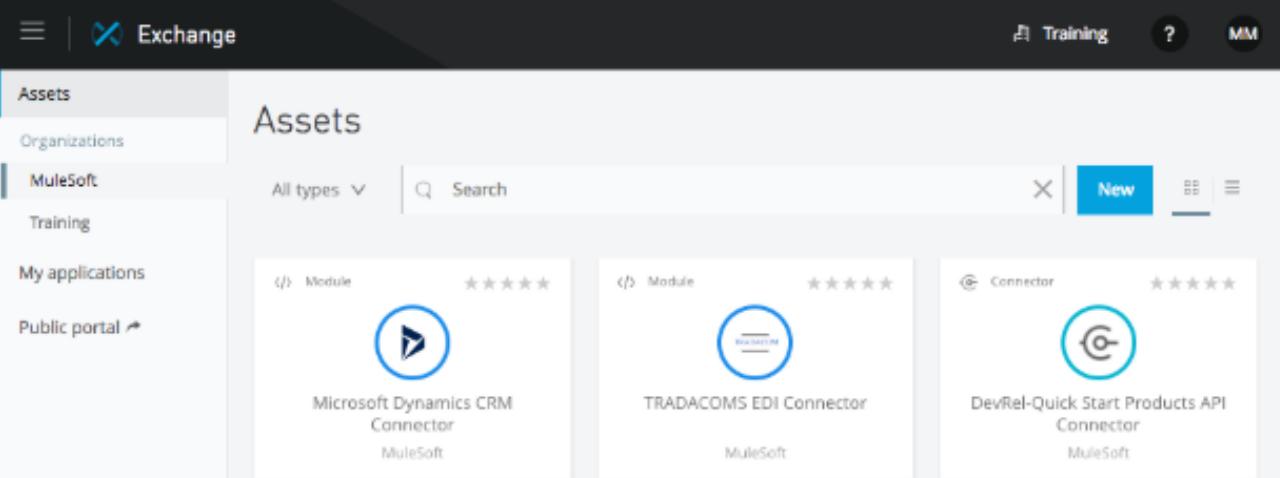
The screenshot shows the Runtime Manager interface. At the top, there is a navigation bar with icons for Training, Help, and MM. Below the navigation bar is a sidebar with the following menu items: DESIGN (selected), Applications (highlighted in blue), Servers, Alerts, VPCs, and Load Balancers. The main content area features a large, stylized gray icon of a head with two ears. Below the icon, the text "There are no applications to show" is displayed. At the bottom of the main area is a button labeled "Deploy application".

11. In the main menu, select API Manager.

The screenshot shows the API Manager interface. At the top, there is a navigation bar with icons for Training, Help, and MM. Below the navigation bar is a sidebar with the following menu items: SANDBOX (selected), API Administration (highlighted in blue), Client Applications, Custom Policies, and Analytics. The main content area features a large, stylized gray icon of a chart. Below the icon, the text "No APIs to display. Get started by adding your first API." is displayed. At the bottom right of the main area is a button labeled "Select an API version to see more details".

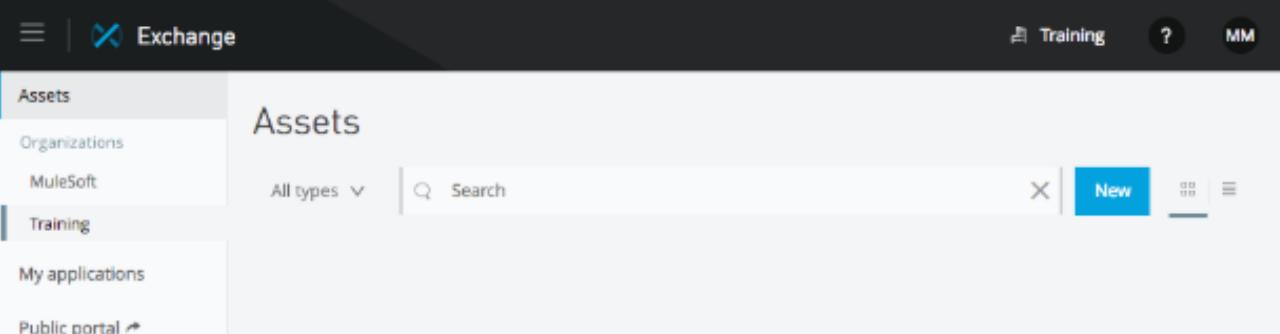
Explore Anypoint Exchange

12. In the main menu, select Exchange.
13. In the left-side navigation, select MuleSoft; you should see all the content in the public Exchange.



The screenshot shows the Anypoint Exchange interface. The top navigation bar includes 'Training', a question mark icon, and a 'MM' button. The left sidebar has sections for 'Assets', 'Organizations', 'MuleSoft' (which is selected), 'Training', 'My applications', and 'Public portal'. The main content area is titled 'Assets' and shows a search bar with 'All types' dropdown and a 'Search' input field. There is a 'New' button and a settings icon. Below the search bar, there are three items listed: 'Module' (Microsoft Dynamics CRM Connector by MuleSoft), 'Module' (TRADACOMS EDI Connector by MuleSoft), and 'Connector' (DevRel-Quick Start Products API Connector by MuleSoft). Each item has a star rating of five stars.

14. In the left-side navigation, select 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.



This screenshot shows the same Anypoint Exchange interface, but with a different selection in the left sidebar. The 'Training' section is now selected. The rest of the interface is identical to the previous screenshot, showing the 'Assets' section with the same search bar, 'New' button, and three connector items.

15. In the left-side navigation, select MuleSoft.

16. In the types menu, select Connectors.

The screenshot shows the Exchange interface with the following details:

- Header:** Exchange
- Left Navigation:** All, MuleSoft, Training, My applications, Public Portal
- Toolbar:** Training, ?, MM
- Section:** All assets
- Search:** Connectors ▾, Search
- Actions:** X, New, More
- Items:** Three connectors are listed:
 - LDAP Connector** (MuleSoft) - Rated 5 stars
 - Amazon RDS Connector** (MuleSoft) - Rated 5 stars
 - Amazon EC2 Connector** (MuleSoft) - Rated 5 stars

17. Select one of the connectors and review its information.

18. In the left-side navigation, click the Assets list link.

19. Search for the Salesforce Connector and review its details.

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

The screenshot shows the MuleSoft Anypoint Exchange interface. The top navigation bar includes 'Assets list' (highlighted in blue), 'Exchange' (with a gear icon), 'Training', a question mark icon, and a 'MM' icon. On the left, a sidebar has 'Assets list' and 'Salesforce Connector' (highlighted in blue) under 'Helpful Links'. The main content area displays the 'Salesforce Connector' product page. It features a blue circular icon with a white cloud, the product name 'Salesforce Connector', a 5-star rating with '(0 reviews)', and a 'Rate and review' link. A detailed description follows: 'The Anypoint Salesforce Connector enables businesses to create instant connectivity between Salesforce and popular ERP, analytics, billing, marketing automation, social applications, and services.' Below this is another section: 'MuleSoft's Salesforce Integration solutions, powered by the Anypoint™ Platform, allow organizations to do more throughout the enterprise, simplifying business processes, thereby allowing a greater focus on core business requirements and less time on overcoming integration challenges.' A note states: 'Note: This connector module only works with Mule 4 and Studio 7.' A 'Supported Salesforce APIs' section lists compatibility with Design Center and various Salesforce API versions (v37-v41). A table at the bottom shows the 'Version' (9.1.0) and 'Runtime version' (4.1.0). The right side of the interface shows sections for 'Download', 'Dependency Snippets', 'Overview' (listing 'Type: Module', 'Created By: MuleSoft Organization', and 'Published On: Feb 21, 2018'), and 'Versions'.

20. In the left-side navigation, click Assets list.

21. In the types menu, select Templates.

22. Remove salesforce from the search field and press Enter/Return.

Browse REST APIs in Anypoint Exchange

23. In the types menu, select REST APIs.
24. Browse the APIs.

The screenshot shows the Anypoint Exchange interface. At the top, there's a navigation bar with 'Training' and a help icon. Below it is a search bar with 'All assets' and a dropdown set to 'REST APIs'. A 'New' button is also visible. On the left, a sidebar lists categories: 'All', 'MuleSoft', 'Training', 'My applications', and 'Public Portal'. The main area displays three REST API cards: 'Optymyze API' (MuleSoft), 'CardConnect REST API' (MuleSoft), and 'Nexmo SMS API' (MuleSoft). Each card includes a circular logo, the API name, its rating (five stars), and the provider name.

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

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

This screenshot shows a detailed view of the 'Training: American Flights API' card. It features a large green circular icon with a white house-like symbol. Above the icon, the text 'Training: American Flights API' is displayed, followed by the provider 'MuleSoft' at the bottom.

26. Review the API portal.

The screenshot shows the 'Training: American Flights API' page in the MuleSoft API portal. The top navigation bar includes 'Assets list', 'Exchange' (selected), 'Training', '?', and 'MM'. The main content area has a title 'Training: American Flights API | 1.0' with a 5-star rating (7 reviews) and a 'Rate and review' link. A 'Download' button is also present. On the left, a sidebar shows the API summary, types, resources, and various operations like GET /flights, POST /flights, etc. The 'Files' section lists several RAML files: 'AmericanFlightExample.raml', 'AmericanFlightNoIDExample.raml', 'exchange_modules.raml', and 'american-flights-api.raml'. The right side contains an 'Overview' section with details: Type (REST API), Created By (MuleSoft Organization), Published On (Sep 14, 2017), and Visibility (Public). Below this is a 'Asset versions for 1.0' table with two rows: version 1.0.1 (Mocking Service) and 1.0.0.

Version	Instances
1.0.1	Mocking Service
1.0.0	

27. In the left-side navigation, expand and review the list of available resources.

28. Click the GET link for the /flights resource.

29. On the /flights: Get all flights page, review the information for the optional destination query parameter; you should see the API is similar to the one you explored in the public Muletraining portal.

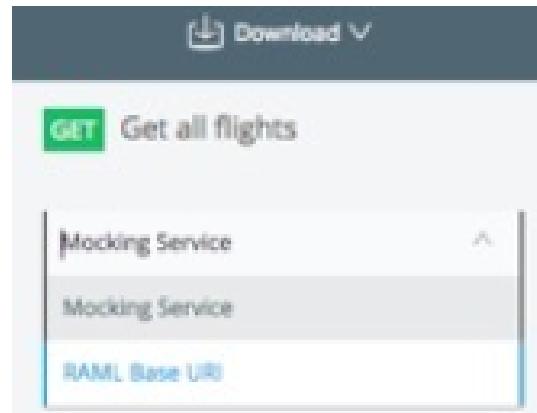
The screenshot shows the Muletrain API Exchange interface. On the left, there's a sidebar with navigation links like 'Assets list', 'Training: American Flights API' (which is highlighted), 'API summary', 'Types', 'Resources', 'Flights' (which is expanded), and 'API instances'. Under 'Flights', several actions are listed: 'Get all flights' (highlighted in green), 'Add a flight' (purple), 'Get a fl...', 'Delete...', and 'Update...'. The main content area displays the 'Training: American Flights API | 1.0' page. It features a home icon, a 5-star rating (7 reviews), and a title 'Get all flights'. Below this, it says '/flights : Get all flights'. A 'Request' section shows a GET method at `http://training-american-ws.cloudhub.io/api/flights`. A 'Parameters' table is shown:

Parameter	Type	Description
destination	string (enum)	Destination airport code Possible values: SFO, LAX, CLE

Below the table, there's a 'Response' section. On the right, a detailed view of the 'Get all flights' endpoint is shown with tabs for 'Mocking Service' (selected), 'Parameters' (selected), and 'Headers'. The 'Parameters' tab shows the 'destination' query parameter with a checkbox for 'Show optional parameters' and a 'Send' button.

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

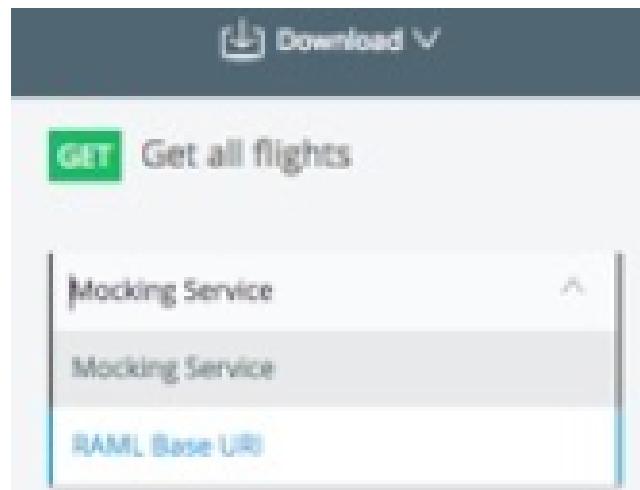
30. In the API console, review the options for the instances you can test.



31. Select Mocking Service.

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

30. In the API console, review the options for the instances you can test.



31. Select Mocking Service.

32. Select a destination and click Send; you should get the two example flights.

The screenshot shows a screenshot of a web-based API testing tool. At the top, a green button labeled "GET" is followed by the text "Get all flights". Below this is a dropdown menu set to "Mocking Service". The URL field contains "https://mocksvc-proxy.anypoint.mulesoft.com/exc". There are two tabs: "Parameters" (which is selected) and "Headers". Under "Parameters", there is a section for "Query parameters" with a dropdown containing "LAX". A checkbox labeled "Show optional parameters" is checked. Below the parameters is a blue "Send" button. The response section shows a green "200 OK" status with a timestamp of "295.75 ms". There is a "Details" link with a dropdown arrow. Below this are four icons: a checkmark, a download arrow, a file icon, and a refresh icon. The response body is displayed in a code block:

```
[Array[2]
-0: {
  "ID": 1,
  "code": "ER38sd",
  "price": 400,
  "departureDate": "2017/07/26",
  "origin": "CLE",
  "destination": "SFO",
  "emptySeats": 0,
```

33. Change the API instance from Mocking Service to RAML Base URI.

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

GET Get all flights

RAML Base URI

http://training-american-ws.cloudhub.io/api/flight:

Parameters Headers

Query parameters Show optional parameters

LAX

Send

200 OK 1310.77 ms Details ↴

📋 ↴ </> ⌂

```
[Array[3]
-0: {
  "ID": 1,
  "code": "rreee001",
  "price": 541,
  "departureDate": "2016-01-20T00:00:00",
  "origin": "MUA",
  "destination": "LAX",
  "airline": "American Airlines"
}
-1: {
  "ID": 2,
  "code": "rreee002",
  "price": 541,
  "departureDate": "2016-01-20T00:00:00",
  "origin": "MUA",
  "destination": "LAX",
  "airline": "American Airlines"
}
-2: {
  "ID": 3,
  "code": "rreee003",
  "price": 541,
  "departureDate": "2016-01-20T00:00:00",
  "origin": "MUA",
  "destination": "LAX",
  "airline": "American Airlines"
}]
```

Building integration applications and APIs with Design Center



Design Center anatomy



Design Center

Training ? MM

Projects

Search...

Name	Project Type	Last Update
American Flights Example	API Fragment	July 27th, 2017
MUA Flights API	API Specification	July 27th, 2017
MUA Flight Data Type	API Fragment	July 27th, 2017
American Flight Data Type	API Fragment	July 27th, 2017
American Flight Example	API Fragment	July 27th, 2017
American Flights App	Mule Application	July 27th, 2017
Training American Flights API	API Specification	July 27th, 2017

Get Started

+ Create

- API Specification
- API Fragment
- Mule Application
- Get Anypoint Studio

American Flights App

Created with API designer

Created with flow designer

Modified July 27th, 2017

Created July 27th, 2017

Created by stallons

Environment 0d6bd95d-e1d2-461d-8b2e-ad0ab31edd64

Status Ready to deploy

Deployment url americanflightsapp-jkb.cloudhub.io

Open

Design Center applications



Application	Purpose	In this course	Additional courses
flow designer	Web app for building integration apps that connect systems and consume APIs	2 WTs	<ul style="list-style-type: none">• Anypoint Platform: Flow Design
API designer	Web app for designing, documenting, and mocking APIs	Module 3	<ul style="list-style-type: none">• Anypoint Platform: API Design
Anypoint Studio	Desktop IDE for implementing APIs and building integration applications	Module 4 In Fundamentals: Modules 6-13	

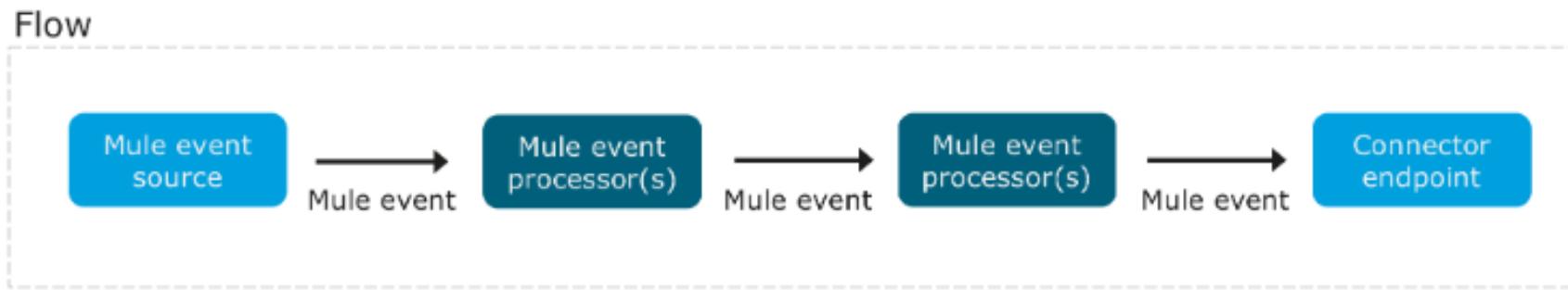
Both flow designer and Anypoint Studio create Mule applications



- **Mule applications** can be created
 - Visually using flow designer or Anypoint Studio
 - By writing code (primarily XML) using Anypoint Studio (or other tools)
- Under the hood, Mule applications are Java applications using Spring
- Mule applications are deployed to a **Mule runtime**
 - Mule runtimes can be MuleSoft-hosted in the cloud (CloudHub) or customer-hosted in the cloud or on-prem

- **A lightweight Java-based enterprise service bus (ESB) and integration platform** that allows developers to connect apps together quickly and easily, enabling them to exchange data
 - Acts as a transit system for carrying data between apps (the Mule)
 - Can connect all systems including web services, JMS, JDBC, HTTP, & more
- **Decouples point-to-point integrations** by having all (non-Mule) apps talk to the bus (to a Mule runtime) instead of directly to each other
- **Can be deployed anywhere**, can integrate and orchestrate events in real time or in batch, and has universal connectivity
- **Enforces policies for API governance**

- Mule applications receive events, process them, and route them to other endpoints
- **Mule applications** accept and process a **Mule event** through a series of **Mule event processors** plugged together in a **flow**

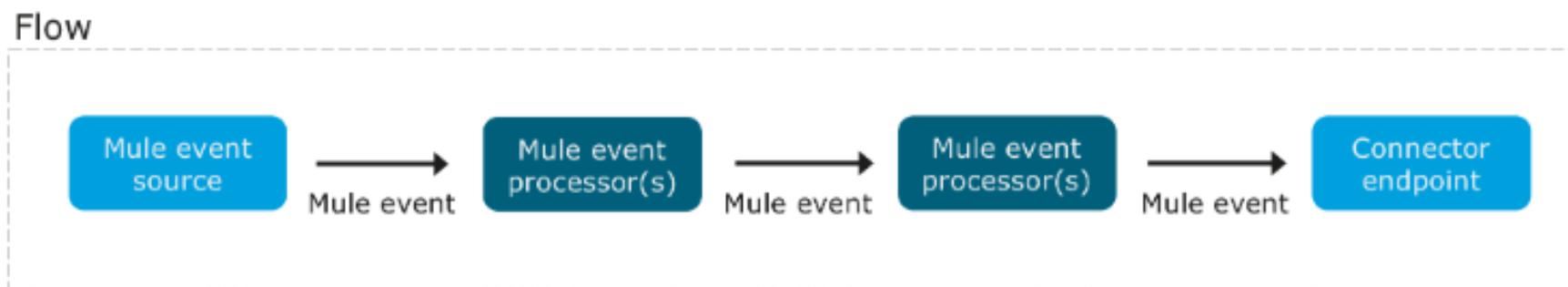


- An application can consist of
 - A single flow
 - Multiple flows
 - Multiple flows connected together

What's in a typical Mule 4 flow?



- A **Mule event source** that initiates the execution of the flow
 - Can be triggered by an event like
 - A consumer request from a mobile device
 - A change to data in a database
 - The creation of a new customer ID in a SaaS application
- **Mule event processors** that transform, filter, enrich, and process the event and its message



Creating integration applications with flow designer



Flow designer anatomy



Design Center > American Flights App

Saved 8 minutes ago Latest changes applied Run americanflightsapp-hvac.c... Running Deploy

Project Get flights Application status

Flows + Project explorer

Get flights Reusable Configu CloudHub HTTP Training: American Flights API

Data Types + Flights

HTTP Listener flights Payload: Object Attributes: HttpRequestAttri...

Training: Americ... Get All Flights Payload: Array of Objects Attributes: HttpRequestAttri...

Transform DataWeave Payload: Flights

Logs Options : Expand Logs ^

* Application: americanflightsapp-hvac
* OS encoding: UTF-8, Mule encoding: UTF-8
*

SYSTEM 06:22:35 Worker(18.220.102.247): Your application has started successfully.
SYSTEM 06:22:35 Your application is started.
INFO 06:22:48 *****

Cards Logs

- When you create a Mule application project in Design Center
 - A new application is created and opened in flow designer
 - **The application is deployed to a MuleSoft-hosted Mule runtime (called a CloudHub worker) in the cloud and started**
- When you make changes to the application in flow designer and are ready to test it
 - You need to run the application again – which updates the application deployed to the worker



MuleSoft-hosted runtime

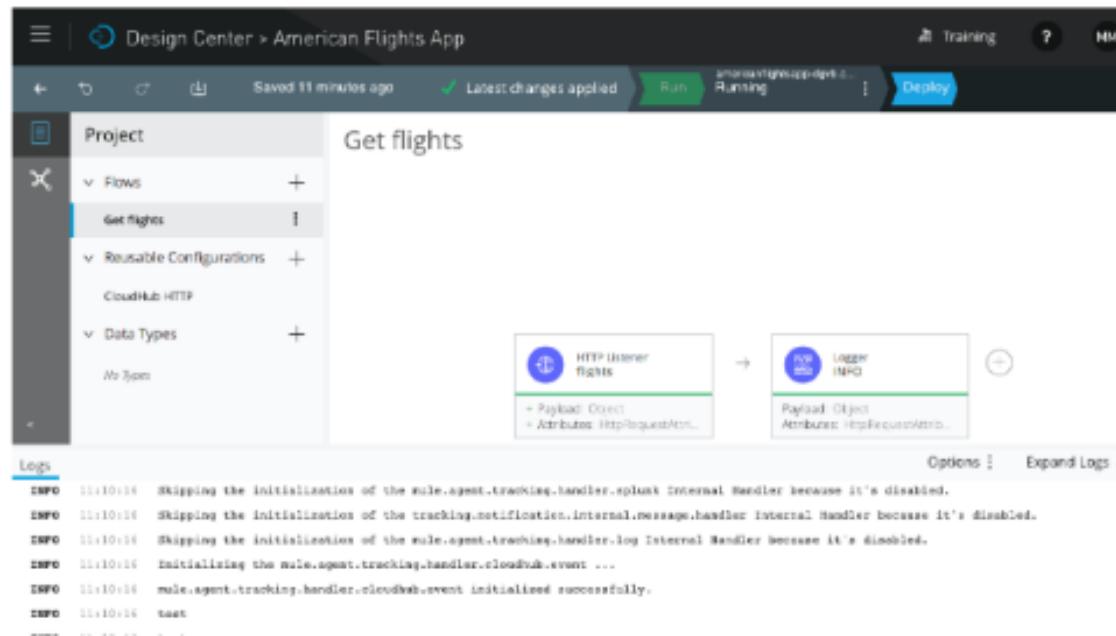
- **A worker is a dedicated instance of Mule that runs an app**
- Each worker
 - Runs in a separate container from every other application
 - Is deployed and monitored independently
 - Runs in a specific worker cloud in a region of the world
- Workers can have a different memory capacity and processing power
 - Apps can be scaled vertically by changing the worker size
 - Apps can be scaled horizontally by adding multiple workers
- There are workers in different environments
 - Design (for flow designer apps only), Sandbox, Production..
 - Apps can be promoted from one environment to another

Worker size
0.1 vCores
0.1 vCores
500 MB memory
0.2 vCores
1 GB memory
1 vCore
1.5 GB memory
2 vCores
3.5 GB memory

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



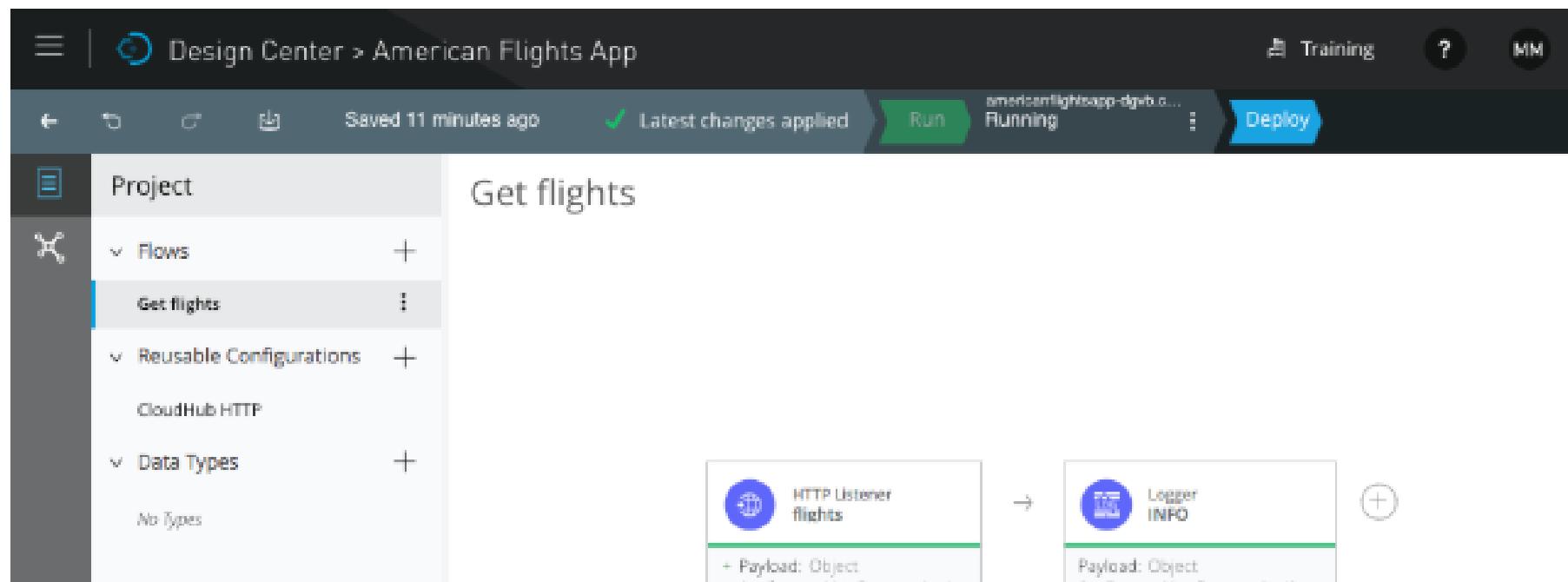
- Create a new Mule application project in Design Center
- Create an HTTP trigger for a flow in the application
- Add a Logger component
- Run and test the application
- View application information in Runtime Manager

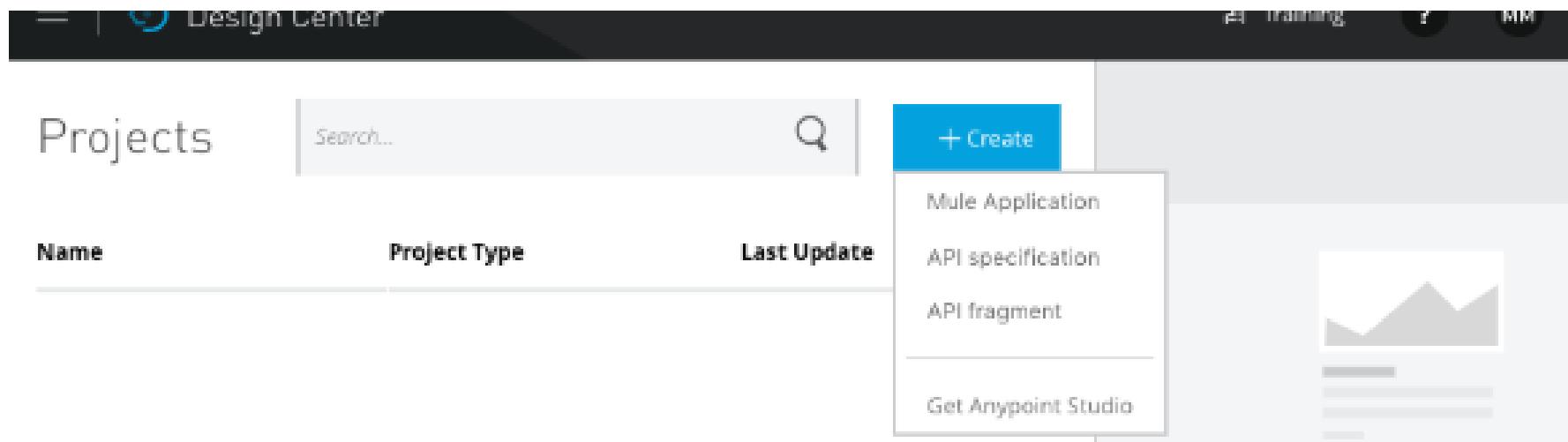


Walkthrough 2-2: Create a Mule application with flow designer

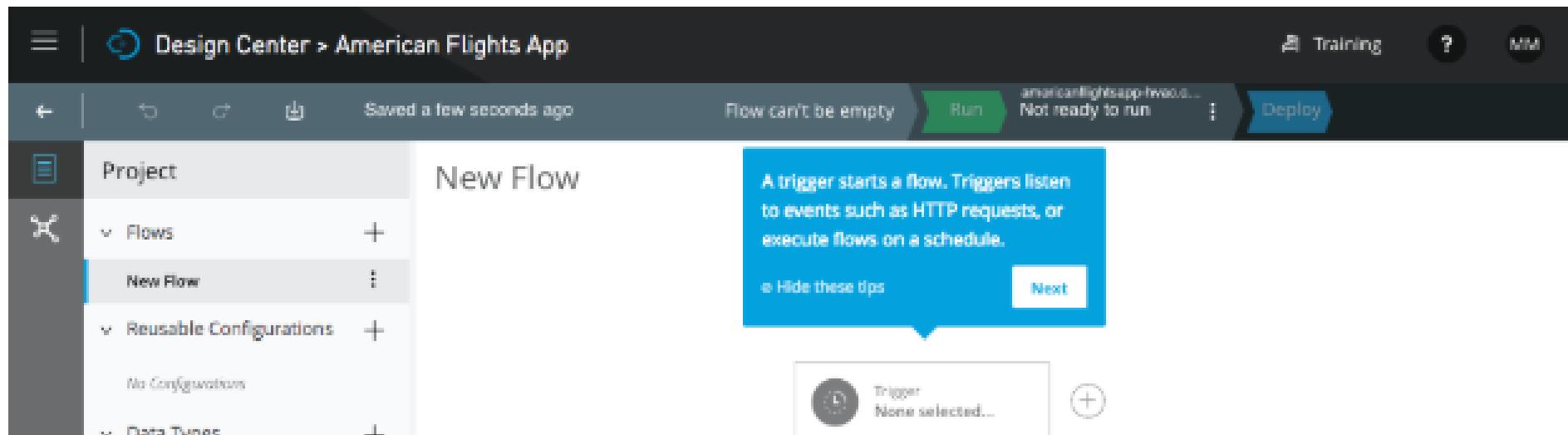
In this walkthrough, you build, run, and test a basic Mule application with flow designer. You will:

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

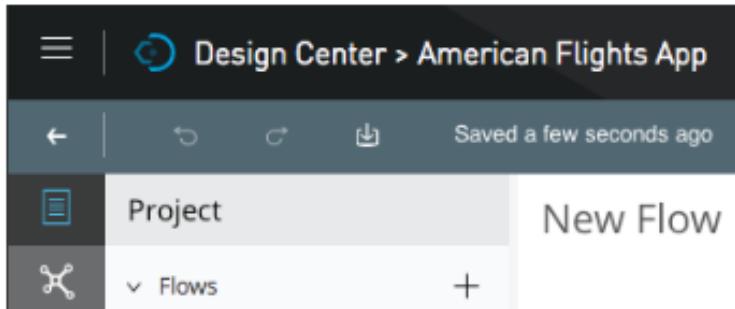




4. In the New Mule Application dialog box, set the project name to American Flights App.
5. Click Create; flow designer should open.



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 page.

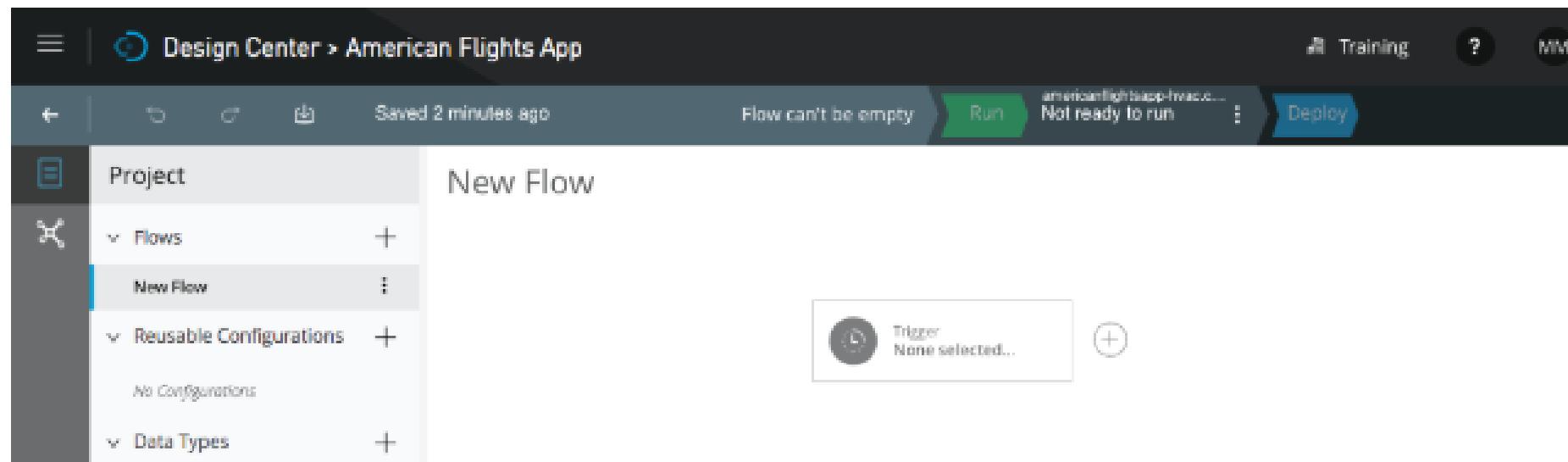
Name	Project Type	Last Update
American Flights App	Mule Application	April 18th, 2018

American Flights App

Details	
Name	American Flights App
Modified	April 18th, 2018
Created	April 18th, 2018
Created by	maxmule4
Environment	6b73b560-e3f8-464d-be77-b6f96a132855
Status	Running
Deployment url	americanflightsapp-hvac.cloudhub.io

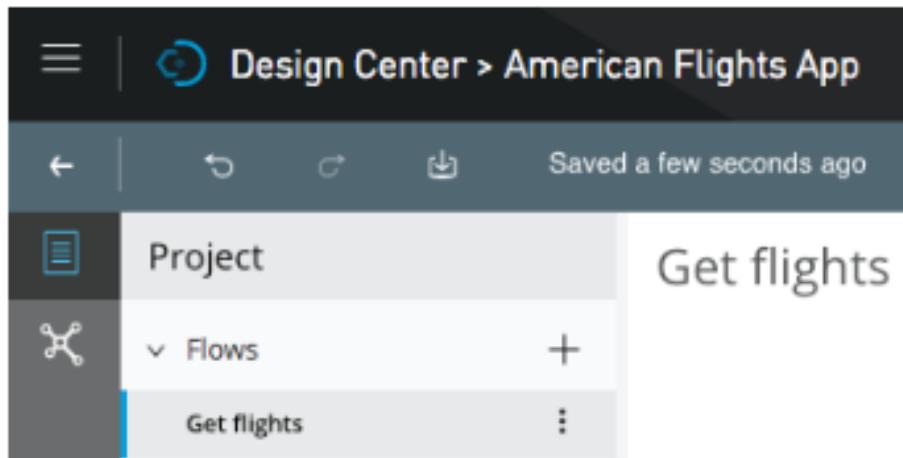
Open

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



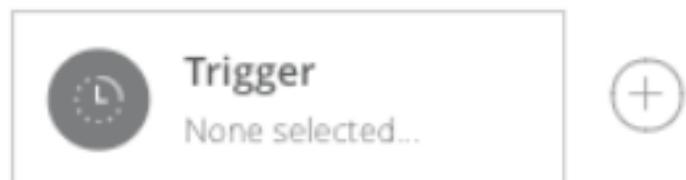
Rename the flow

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.

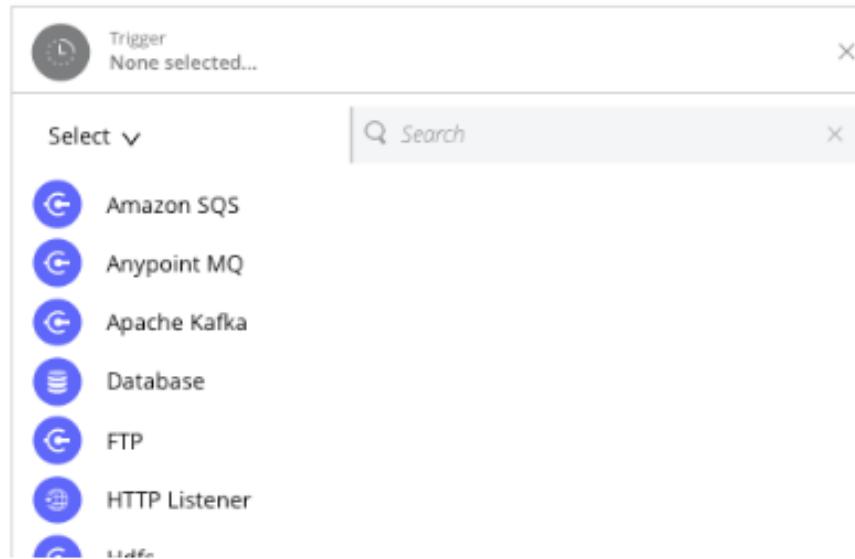


Create an HTTP trigger for a flow in the application

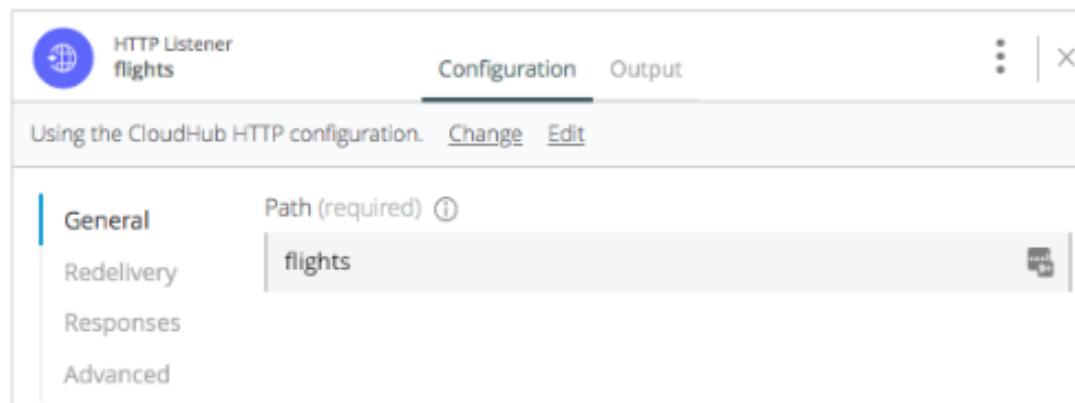
13. In flow designer, click the Trigger card.



14. In the Trigger card, select HTTP Listener.

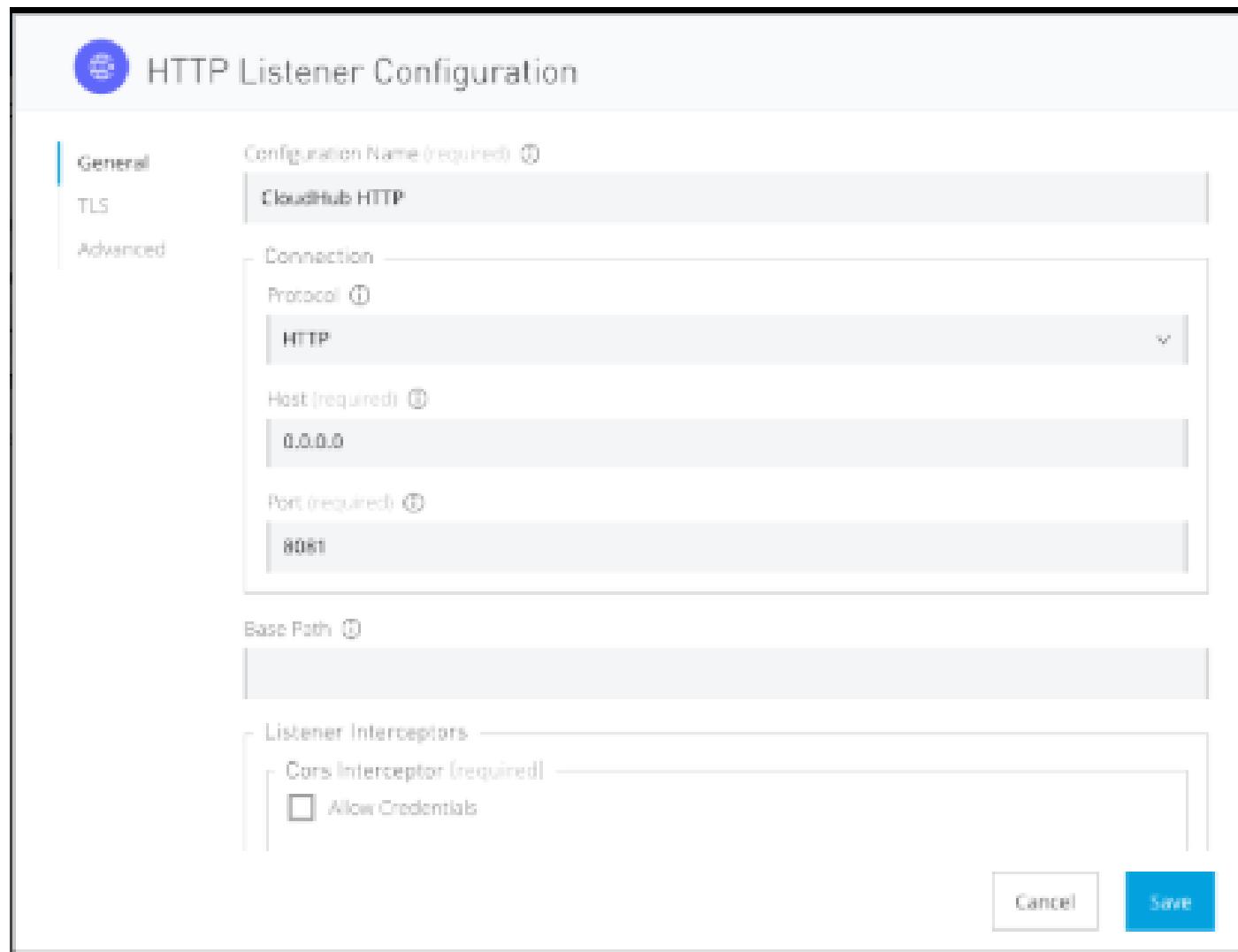


15. In the HTTP Listener dialog box, set the path to flights.

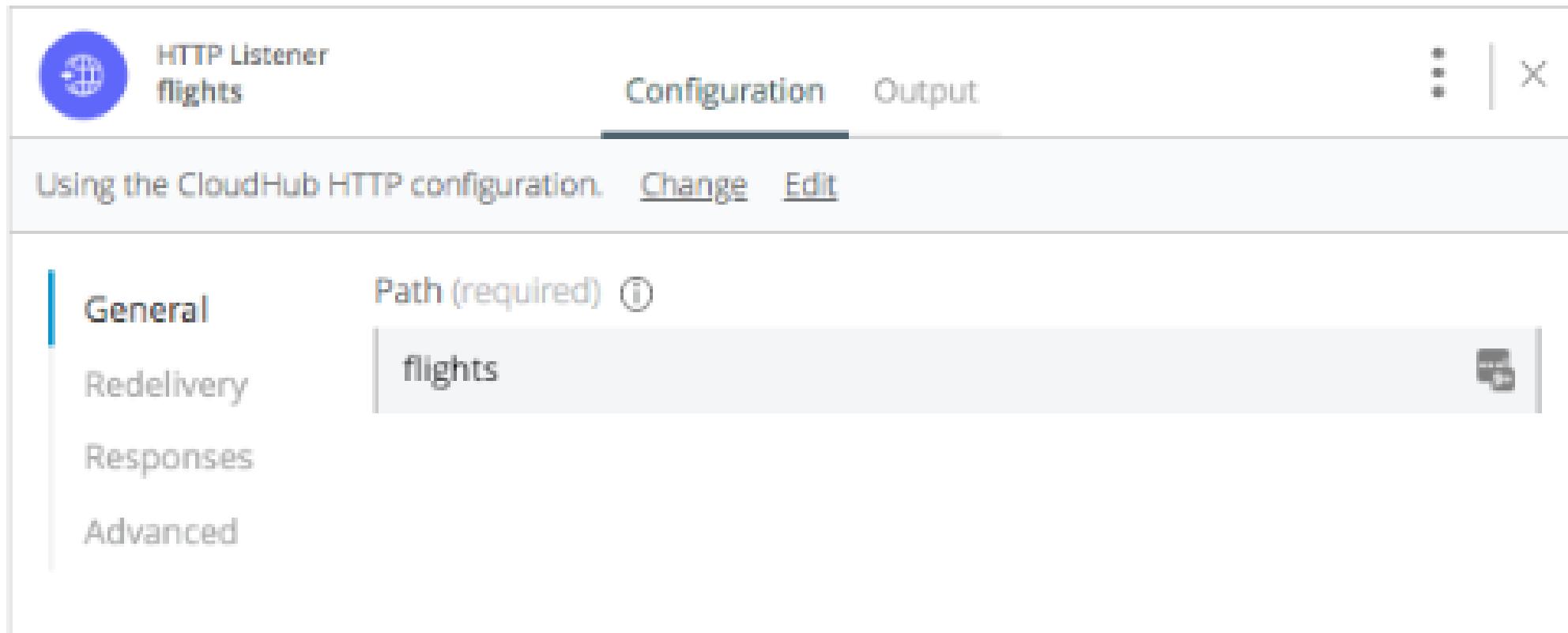


16. Click the Edit link for the CloudHub HTTP configuration.

17. In the **HTTP Listener Configuration** dialog box, review the information and click **Cancel**.

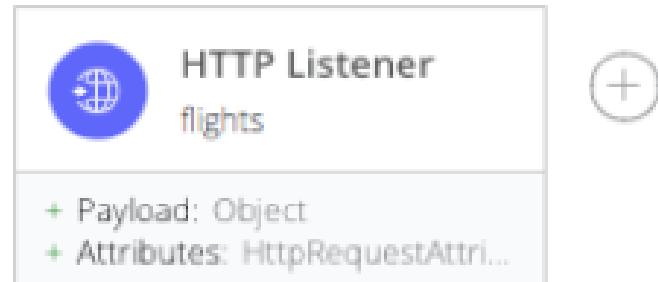


18. In the HTTP Listener dialog box, click the close button in the upper-right corner.

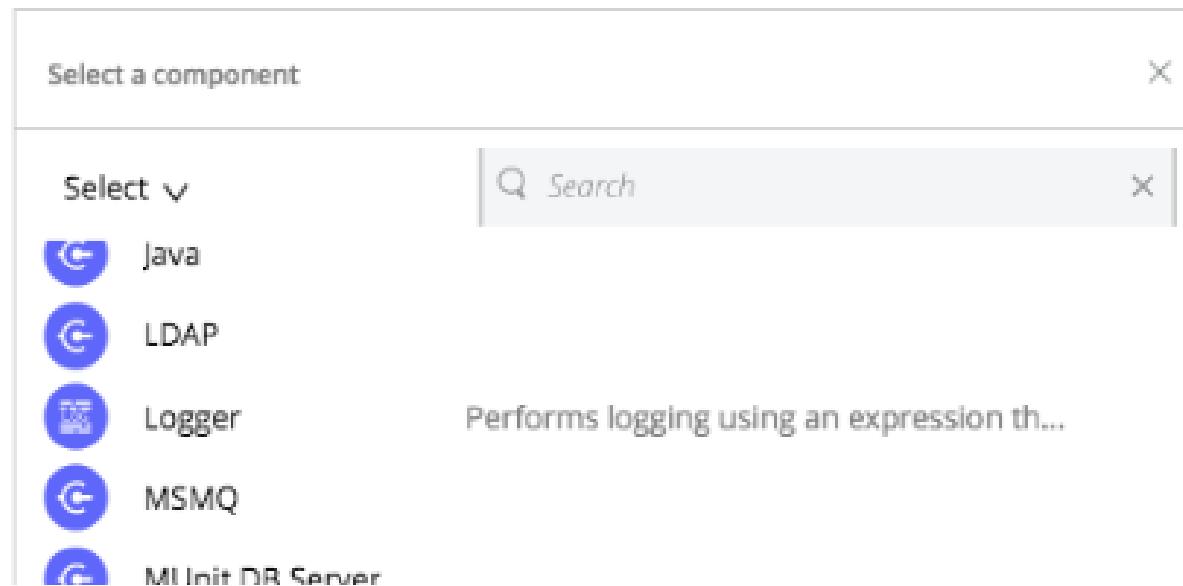


Add a Logger

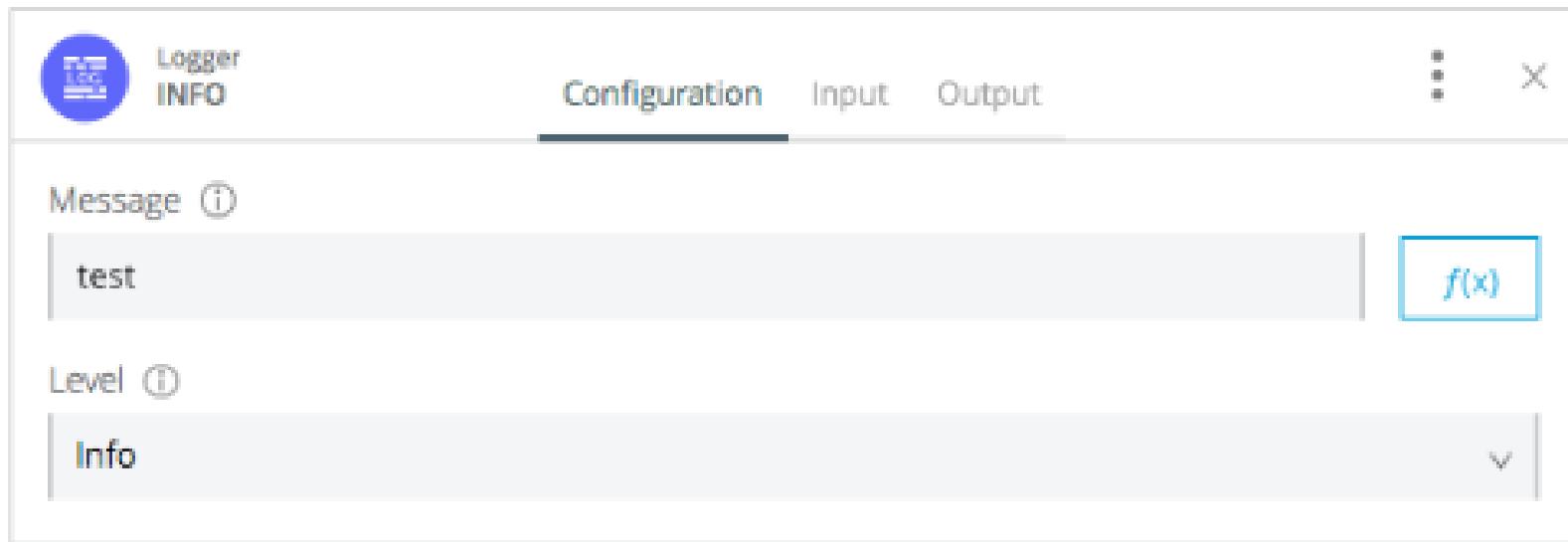
19. Click the add button next to the HTTP Listener card.



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 the middle of both cards.



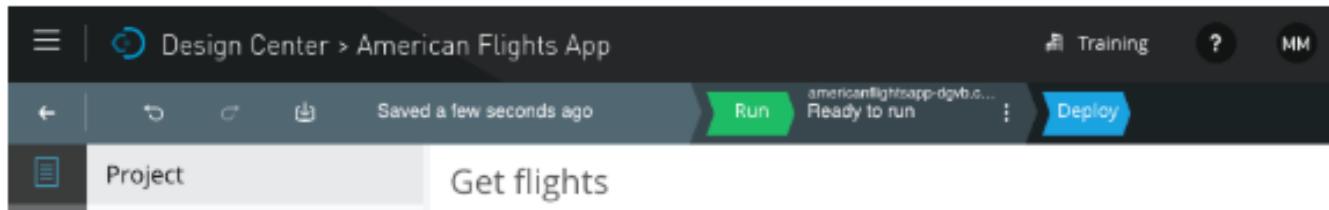
Deploy the application

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

The screenshot shows the Mule Studio interface with the 'Logs' tab selected. At the top, there are two panels: 'Data Types' on the left and 'Logger' on the right. The 'Logger' panel shows two log entries: 'HTTP Listener flights' and 'Logger INFO'. Below the logs, the terminal output is displayed:

```
INFO 10:59:11 ****
* Application: americanflightsapp-dgub
* OS encoding: UTF-8, Mule encoding: UTF-8
*
*****
SYSTEM 10:59:12 Worker(54.173.168.187): Your application has started successfully.
SYSTEM 10:59:12 Your application is started.
```

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

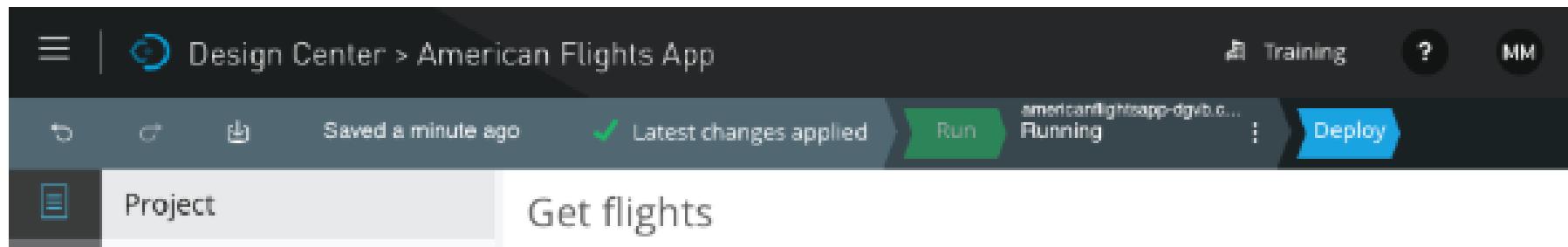


26. Look at the generated URL for the application.

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

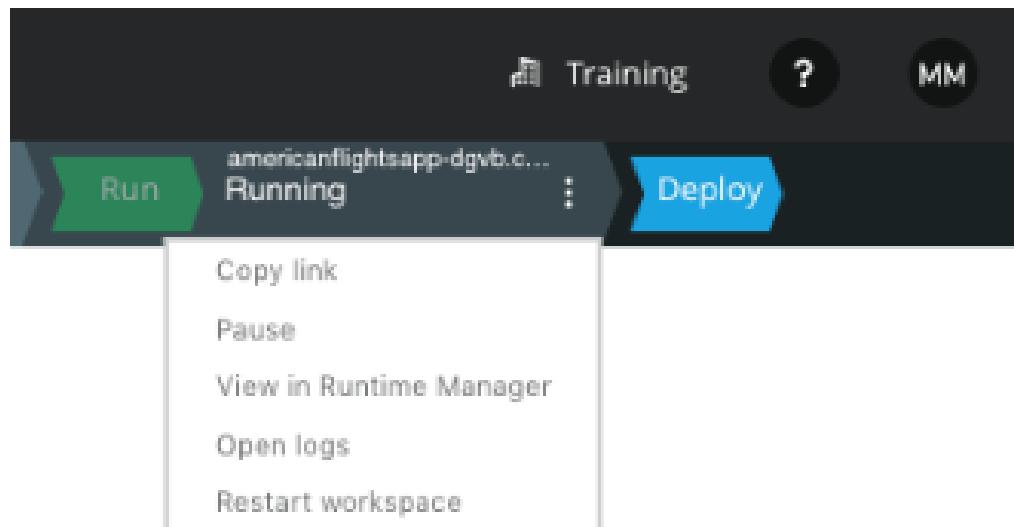
27. Click the Run button.

28. Watch the application status; it should change from Ready to run to Starting application to Running.



Note: If your application fails to run, look at the messages in the Logs panel. If there is no message in the Logs panel, click 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

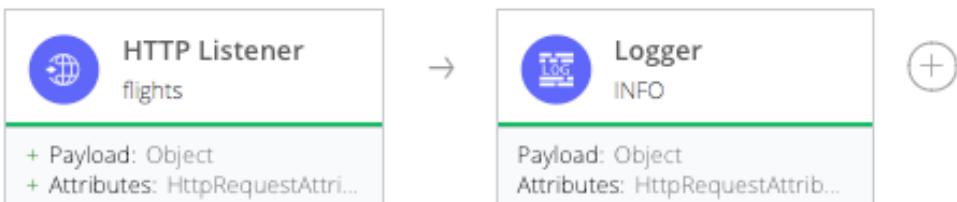
29. Return to Advanced REST Client, paste the copied link, and click Send; you should get a 404 Not Found status with a No listener for endpoint!

The screenshot shows the Advanced REST Client interface. At the top, the method is set to GET and the request URL is http://americanflightsapp-dgzb.cloudhub.io/. Below this, there is a 'Parameters' dropdown. The main response area shows a red box containing '404 Not Found' and '184.90 ms'. To the right is a 'DETAILS' button. Below the response, there are several icons: a square, a triangle, a double-headed arrow, and a circle. A message at the bottom states 'No listener for endpoint: /'.

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

The screenshot shows the Advanced REST Client interface. The request URL is now http://americanflightsapp-dgzb.cloudhub.io/flights. The response area shows a green box containing '200 OK' and '286.15 ms'. To the right is a 'DETAILS' button.

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

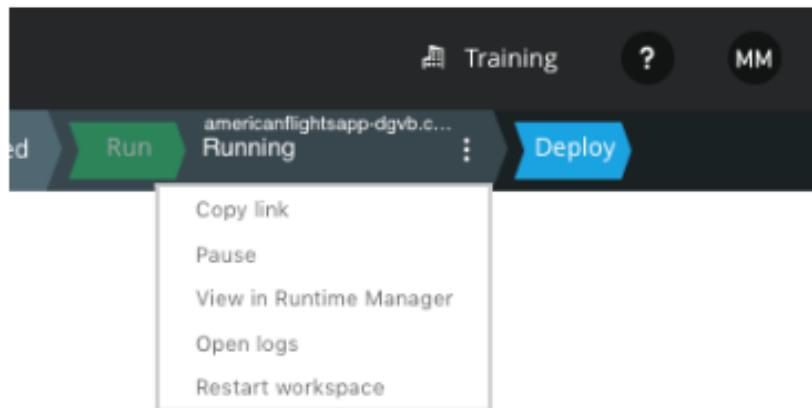


34. Look at the logs; you should see your Logger message displayed twice.

Logs			Options	Expand Logs ^
INFO	11:10:16	Skipping the initialization of the mule.agent.tracking.handler.splunk Internal Handler because it's disabled.		
INFO	11:10:16	Skipping the initialization of the tracking.notification.internal.message.handler Internal Handler because it's disabled.		
INFO	11:10:16	Skipping the initialization of the mule.agent.tracking.handler.log Internal Handler because it's disabled.		
INFO	11:10:16	Initializing the mule.agent.tracking.handler.cloudhub.event ...		
INFO	11:10:16	mule.agent.tracking.handler.cloudhub.event initialized successfully.		
INFO	11:10:16	test		
INFO	11:12:13	test		

View the application in Runtime Manager

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



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

A screenshot of the Runtime Manager interface. On the left, a sidebar has 'DESIGN' selected, and 'Logs' is also visible. The main area shows the application 'americanflightsapp-dgzb' and its 'Live Console' log output. The log output shows several INFO messages from the 'Worker-0' thread, such as skipping initialization of certain handlers because they are disabled. To the right, a 'Deployments' panel shows a deployment log for '10:59 - Deployment' with a 'System Log' entry for 'Worker-0'.

38. 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

The screenshot shows the Mule Runtime Manager interface. The top navigation bar includes 'Training', a help icon, and a 'MM' icon. The main area displays the application 'americanflightsapp-dgzb'. On the left, a sidebar lists 'DESIGN', 'Applications' (selected), 'Dashboard', 'Insight', 'Logs', 'Object Store', 'Queues', and 'Schedules'. The main content area shows the application file as 'americanflightsapp-dgzb.jar', with options to 'Choose file' or 'Get from sandbox' and a 'Stop' button. Below this, 'Last Updated' is listed as '2018-03-25 10:59:12AM' and the 'App url' is 'americanflightsapp-dgzb.cloudhub.io'. A table at the bottom provides runtime details: 'Runtime version' is '4.1.0', 'Worker size' is '0.2 vCores', and 'Workers' is '1'.

	Runtime	Properties	Insight	Logging	Static IPs
Runtime version	4.1.0	0.2 vCores		Workers	1

Accessing, querying, and transforming data





- The data that passes through flows in the app
- Metadata contained in the message header
- The core info of the message - the data the app processes
- Metadata for the Mule event - can be defined and referenced in the app processing the event

- DataWeave 2.0 is the expression language for Mule to access, query, and transform Mule 4 event data
- A JSON-like language that's built just for data query and transformation use cases
 - Full-featured and fully native framework
- Fully integrated with flow designer (and Anypoint Studio)
 - Graphical interface with payload-aware development



The Transform component



- Has input, output, and preview sections with both drag-and-drop and script editors

Select a component

- All
- Salesforce
- ServiceNow
- Transform**
- Try
- Validation

Transform

Configuration Input Output X

Input

- Payload Array<Object>
- > plane Object?
- code String?
- price Number?
- origin String?
- destination String?
- ID Number?
- departureDate String?
- emptySeats Number?

Attributes Void

Variables Object

Output payload

- > Payload Array<Object> (Flight)
- f(x) ○
- airline String?
- flightCode String?
- fromAirportCode String?
- toAirportCode String?
- departureDate String?
- emptySeats Number?
- totalSeats Number?
- price Number?
- planeType String?

Preview

```
1  {
2   "flightCode": "ER38sd",
3   "fromAirportCode": "MIA",
4   "toAirportCode": "SFO",
5   "departureDate": "2016/03/20",
6   "emptySeats": 8,
7   "totalSeats": 150,
8   "price": 400,
9   "planeType": "Boeing 737",
10  "airline": "American"
11 },
12 {
13   "flightCode": "ER45if",
14   "fromAirportCode": "MIA",
15   "toAirportCode": "LAX",
16   "departureDate": "2016/02/11",
17   "emptySeats": 52,
18   "totalSeats": 388,
19   "price": 345.99,
20   "planeType": "Boeing 777",
21   "airline": "American"
22 }
```

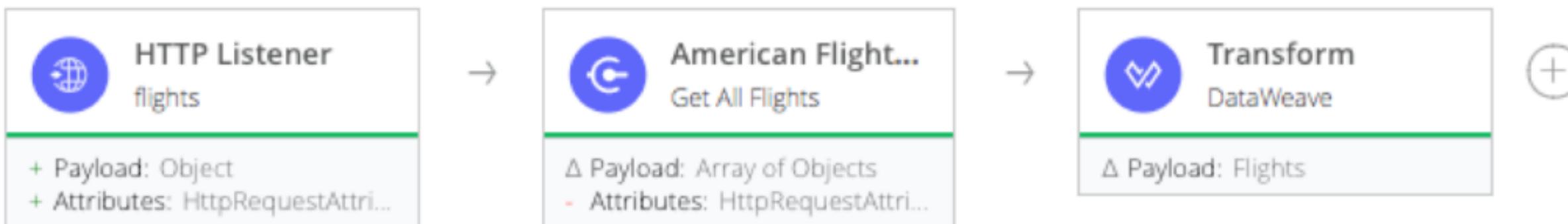
Actions for (root)/#payload

Sample data Script Mappings

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



- 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



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

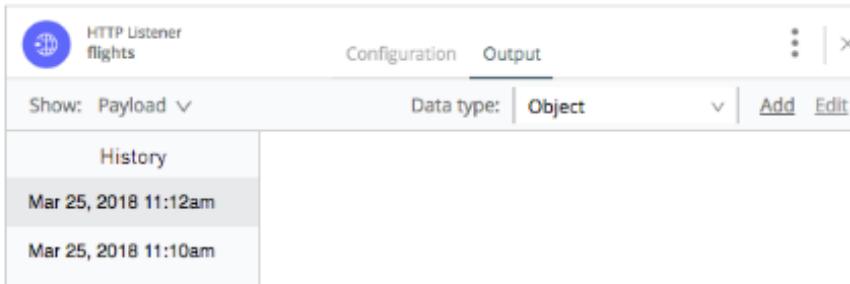
In this walkthrough, you build an integration application to consume an API from Anypoint Exchange. You will:

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



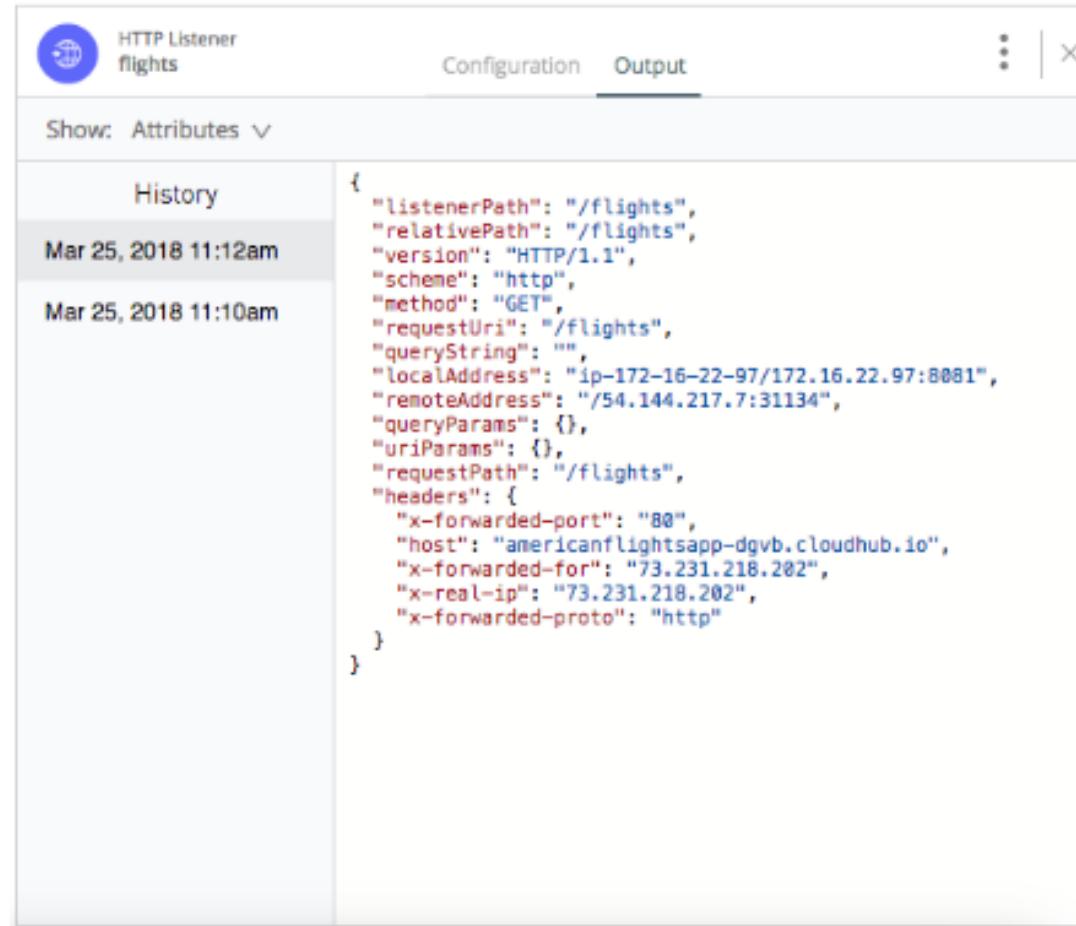
Review Mule event data for the calls to the application

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. Select 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.



The screenshot shows a card for an 'HTTP Listener flights' component. The card has tabs for 'Configuration' and 'Output'. The 'Output' tab is selected, displaying the event attributes. The 'History' section shows two entries: 'Mar 25, 2018 11:12am' and 'Mar 25, 2018 11:10am'. The details for the most recent entry are shown in the main area:

```
{  
    "listenerPath": "/flights",  
    "relativePath": "/flights",  
    "version": "HTTP/1.1",  
    "scheme": "http",  
    "method": "GET",  
    "requestUri": "/flights",  
    "queryString": "",  
    "localAddress": "ip-172-16-22-97/172.16.22.97:8081",  
    "remoteAddress": "/54.144.217.7:31134",  
    "queryParams": {},  
    "uriParams": {},  
    "requestPath": "/flights",  
    "headers": {  
        "x-forwarded-port": "80",  
        "host": "americanflightsapp-dgvb.cloudhub.io",  
        "x-forwarded-for": "73.231.218.202",  
        "x-real-ip": "73.231.218.202",  
        "x-forwarded-proto": "http"  
    }  
}
```

9. Close the card.
10. Open the Logger card.
11. Select the Input tab.

12. Review the payload and attributes values for the two calls.

The screenshot shows a logger interface with the following details:

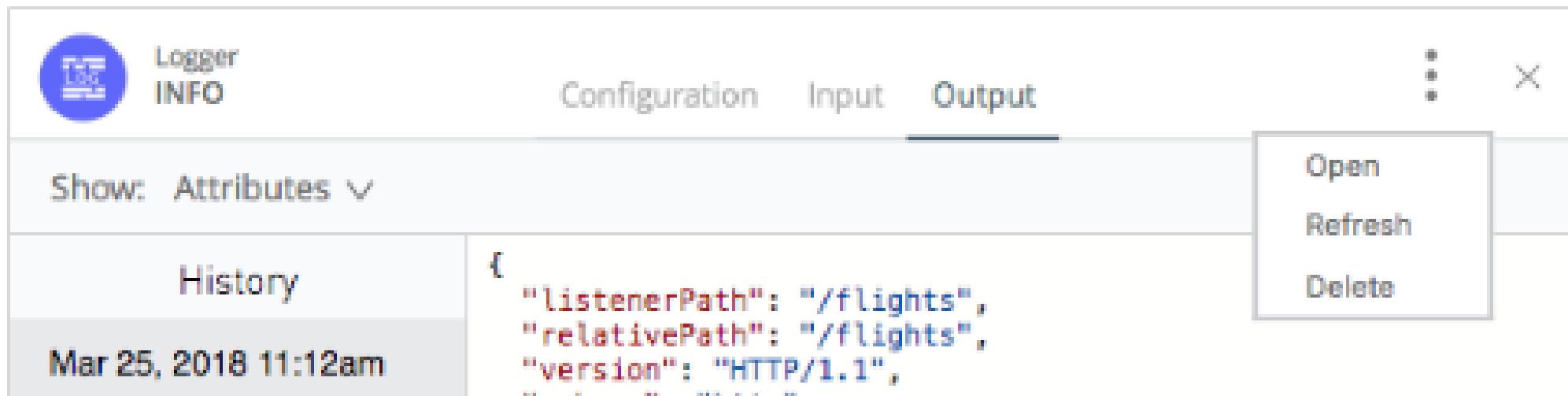
- Logger:** INFO
- Configuration:** (tab)
- Input:** (tab, currently selected)
- Output:** (tab)
- Show:** Attributes ▾
- History:** (Table)
 - Mar 25, 2018 11:12am** (Selected)
 - Mar 25, 2018 11:10am**
- Payload (Selected Row):**

```
{  
  "listenerPath": "/flights",  
  "relativePath": "/flights",  
  "version": "HTTP/1.1",  
  "scheme": "http",  
  "method": "GET",  
  "requestUri": "/flights",  
  "queryString": "",  
  "localAddress": "ip-172-16-22-97/172.16.22.97:8081",  
  "remoteAddress": "/54.144.217.7:31134",  
  "queryParams": {},  
  "uriParams": {},  
  "requestPath": "/flights",  
  "headers": {  
    "Content-Type": "application/json"  
  }  
}
```

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

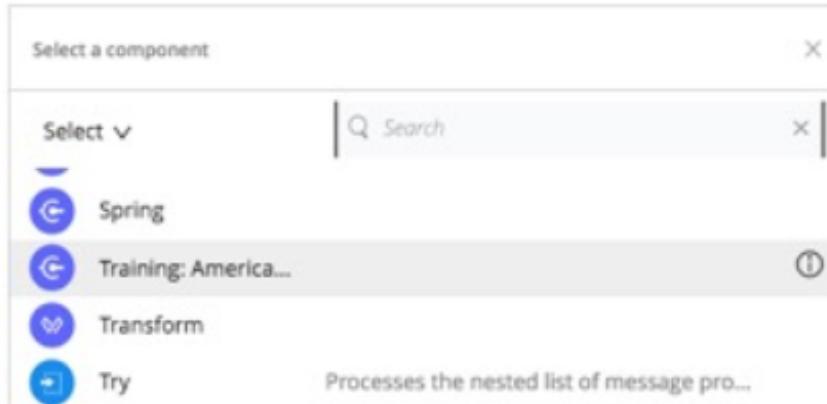
Delete a card

14. Click the options menu for the card and select Delete.

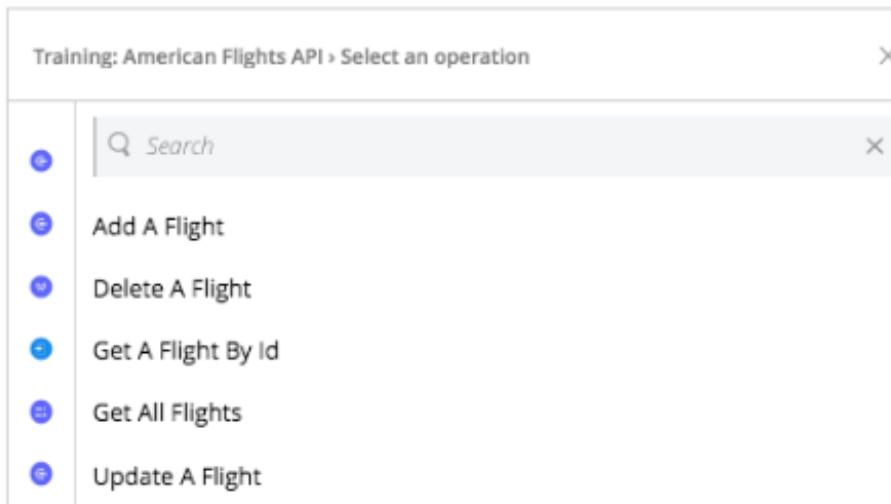


Use the American Flights API in Anypoint Exchange to get all flights

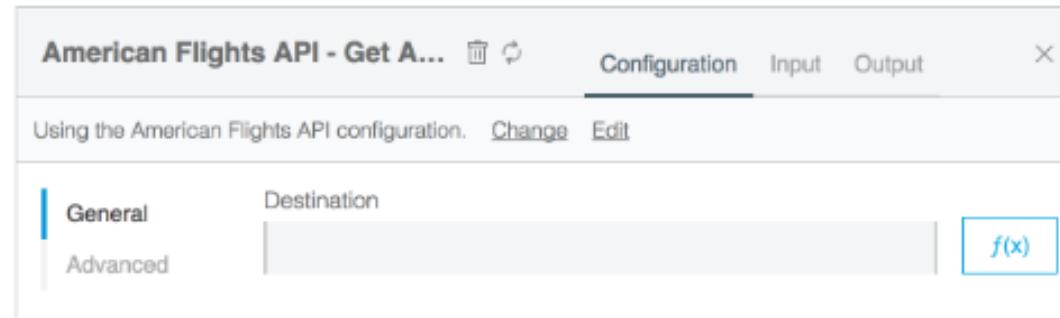
15. Click the Add button next to the HTTP Listener card.
16. In the Select a component dialog box, select the Training: American Flights API.



17. In the Training: 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 Training: American Flights API configuration.



19. Review the information and click Cancel.

A screenshot of a configuration dialog box titled "Training: American Flights API Configuration". The "General" tab is selected. The form fields are as follows:

- Configuration Name (required): Training: American Flights API
- Host: training-american-ws.cloudbuild.io
- Port: 80
- Base Path: /api
- Protocol: HTTP

At the bottom right are "Cancel" and "Save" buttons, with "Save" being highlighted.

20. Close the American Flights API card.



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

22. Wait until the application is running.

Test the application

23. Return to Advanced REST Client and click Send; you should see flight data.

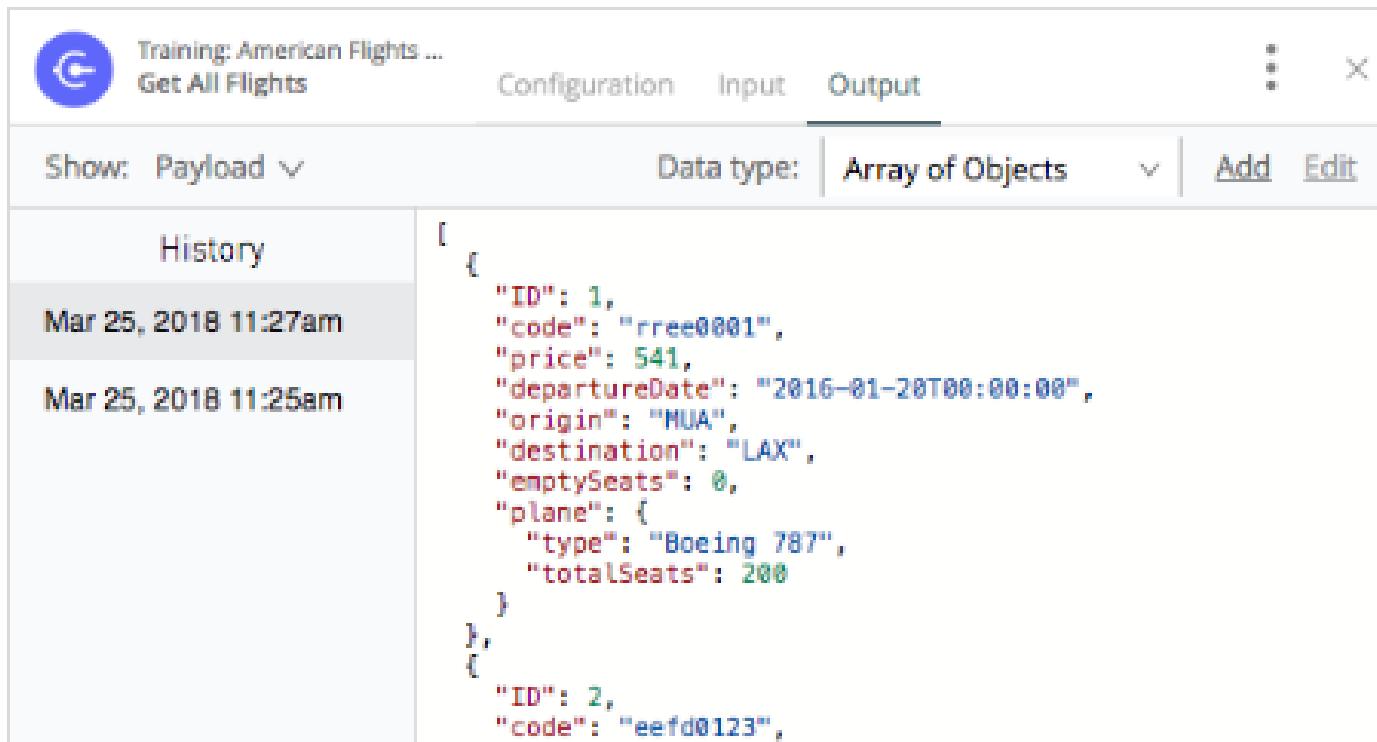
The screenshot shows the Advanced REST Client interface. At the top, the method is set to 'GET' and the URL is 'http://americanflightsapp-dgzb.cloudhub.io/flights'. Below this, there is a 'Parameters' dropdown. The response status is shown as '200 OK' with a green background and '5608.13 ms' latency. To the right is a 'DETAILS' button. Below the status, there are several icons: a copy icon, a refresh icon, a comparison icon, a list icon, and a search icon. The response body is displayed as a JSON array:

```
[  
  {  
    "ID": 1,  
    "code": "rree0001",  
    "price": 541,  
    "departureDate": "2016-01-20T00:00:00",  
    "origin": "MUA",  
    "destination": "LAX",  
    "emptySeats": 0,  
    "plane": {  
      "type": "Boeing 787",  
      "totalSeats": 200  
    }  
  },  
  {  
    "ID": 2,  
    "code": "eefd0123",  
    "price": 541,  
    "departureDate": "2016-01-20T00:00:00",  
    "origin": "MUA",  
    "destination": "LAX",  
    "emptySeats": 0,  
    "plane": {  
      "type": "Boeing 787",  
      "totalSeats": 200  
    }  
  }]
```

24. Click Send again to make a second request.

Review Mule event data

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



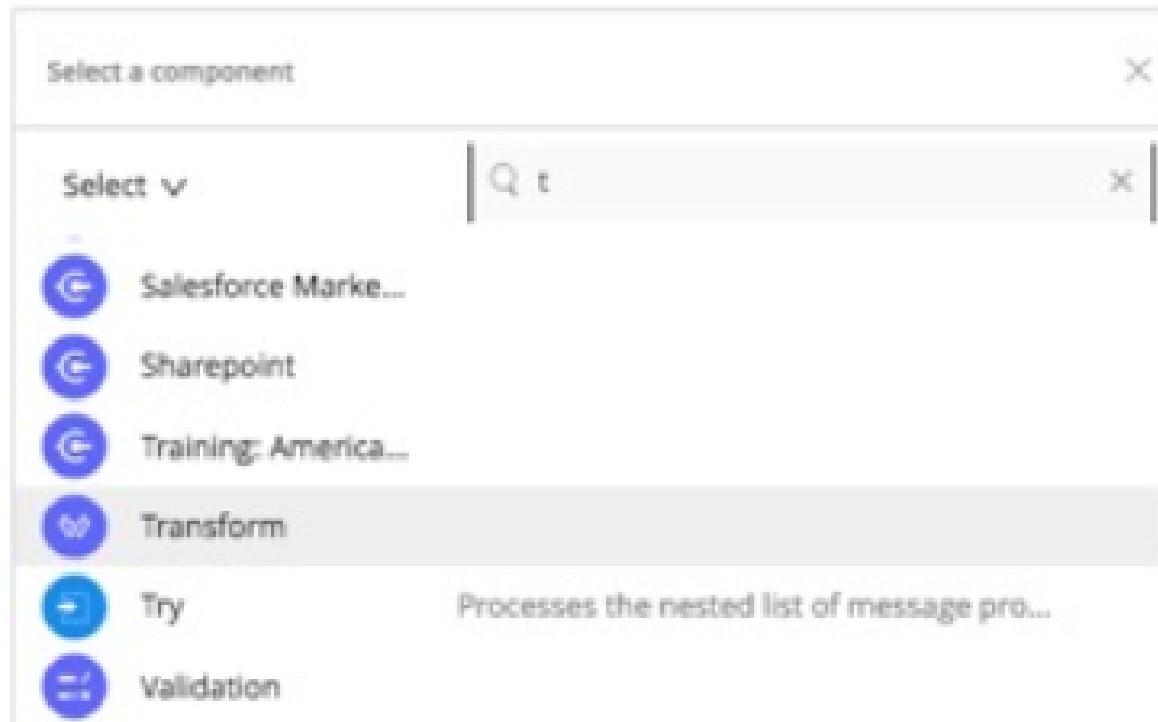
The screenshot shows the Mule API card interface. At the top, there's a blue circular icon with a white arrow pointing right, followed by the text "Training: American Flights ...". Below that is a button labeled "Get All Flights". To the right are tabs for "Configuration", "Input", and "Output", with "Output" being the active tab. Underneath these tabs, there are dropdown menus for "Show: Payload" and "Data type: Array of Objects". On the far right, there are "Add" and "Edit" buttons. The main content area is titled "History" and lists two entries: "Mar 25, 2018 11:27am" and "Mar 25, 2018 11:25am". The payload data is displayed as JSON code. The first entry is fully visible, and the second entry starts with a brace {.

```
[  
 {  
   "ID": 1,  
   "code": "rree0001",  
   "price": 541,  
   "departureDate": "2016-01-20T00:00:00",  
   "origin": "MUA",  
   "destination": "LAX",  
   "emptySeats": 0,  
   "plane": {  
     "type": "Boeing 787",  
     "totalSeats": 200  
   }  
 },  
 {  
   "ID": 2,  
   "code": "eefd0123",  
   "price": 541,  
   "departureDate": "2016-01-20T00:00:00",  
   "origin": "MUA",  
   "destination": "LAX",  
   "emptySeats": 0,  
   "plane": {  
     "type": "Boeing 787",  
     "totalSeats": 200  
   }  
 }]
```

28. Close the card.

Add and configure a component to transform the data

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



31. In the Transform card, look at the Mule event structure in the input section.

32. In the output section, click the Create new Data Type button.

Transform
DataWeave

Configuration Input Output

Input

Output payload

Preview

No data available

Create new Data Type

No data available, please perform some mappings and fill required sample data

Sample data Script Mappings

The screenshot displays the DataWeave Transform interface. The top navigation bar includes 'Transform' and 'DataWeave' buttons, along with 'Configuration', 'Input', and 'Output' tabs. The 'Input' tab is selected, showing a list of fields from a 'payload' object, such as 'plane Object?', 'code String?', 'price Number?', 'origin String?', 'destination String?', 'ID Number?', 'departureDate String?', 'emptySeats Number?', 'attributes Void', and 'vars Object'. The 'Output payload' tab is also selected, showing a tree structure with four items. The 'Preview' tab shows a chart icon and a message indicating 'No data available'. A 'Create new Data Type' button is located in the 'Output payload' section. The bottom navigation bar includes 'Sample data', 'Script', and 'Mappings' tabs, with 'Mappings' being the active tab.

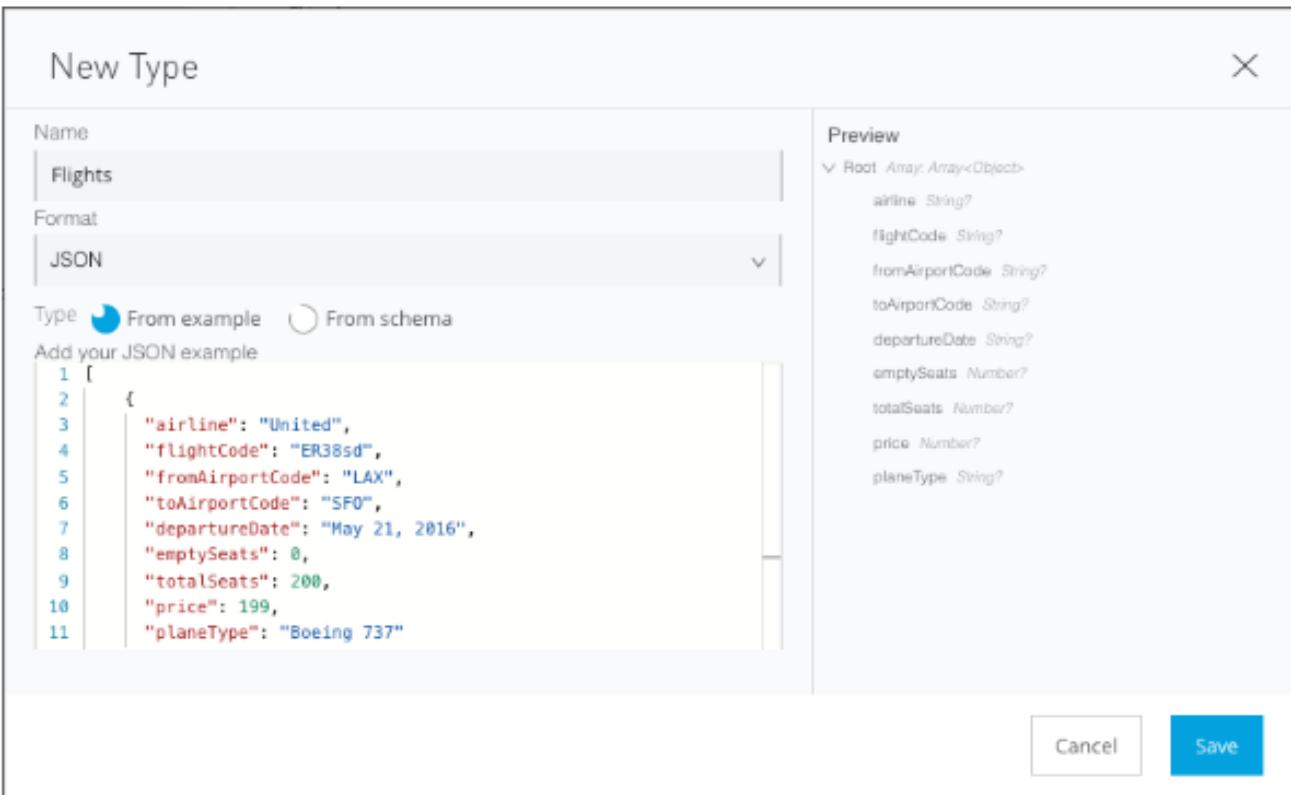
33. In the New Type dialog box, set the following values:

- Name: Flights
- Format: JSON
- Type: From example

34. In the computer's file explorer, return to the student files folder and locate the flights-example.json file in the examples folder.

35. Open the file in a text editor and copy the code.

36. Return to flow designer and paste the code in the section to add your JSON example.



37. Click Save.

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

Transform DataWeave

Configuration Input Output

Input

Output payload

Preview

No data available, please perform some mappings and fill required sample data

Sample data Script Mappings

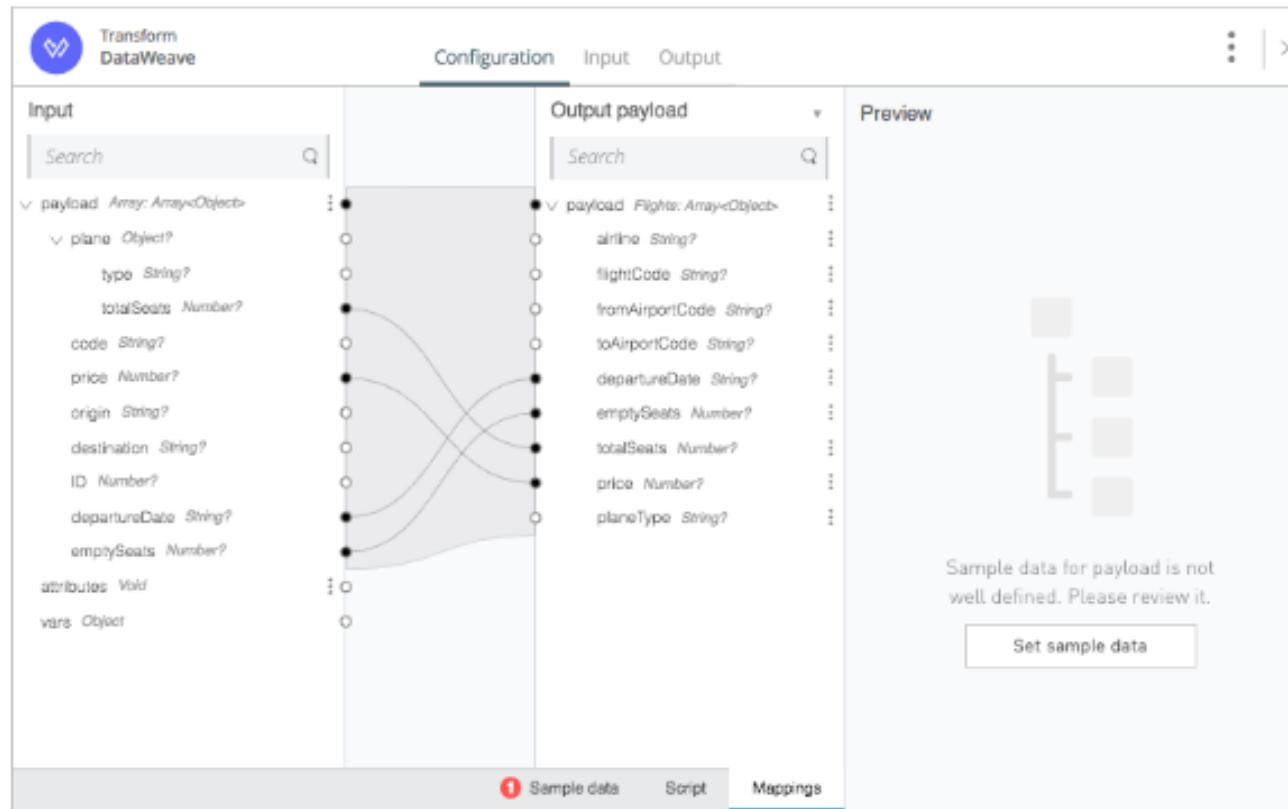
The screenshot shows the DataWeave Transform interface. The top navigation bar includes 'Transform DataWeave' and tabs for 'Configuration', 'Input', and 'Output'. The 'Input' tab is active, showing a search bar and a list of fields and objects. The 'Output payload' section shows a mapped structure. The 'Preview' area contains a small chart and a message: 'No data available, please perform some mappings and fill required sample data'. At the bottom, there are tabs for 'Sample data', 'Script', and 'Mappings', with 'Mappings' being the active tab.

Input	Output payload
payload Array<Object>	v payload Flights: Array<Object>
v plane Object?	o airline String?
o type String?	o flightCode String?
o totalSeats Number?	o fromAirportCode String?
o code String?	o toAirportCode String?
o price Number?	o departureDate String?
o origin String?	o emptySeats Number?
o destination String?	o totalSeats Number?
o ID Number?	o price Number?
o departureDate String?	o planeType String?
o emptySeats Number?	
o attributes Void	
o vars Object	

Create the transformation

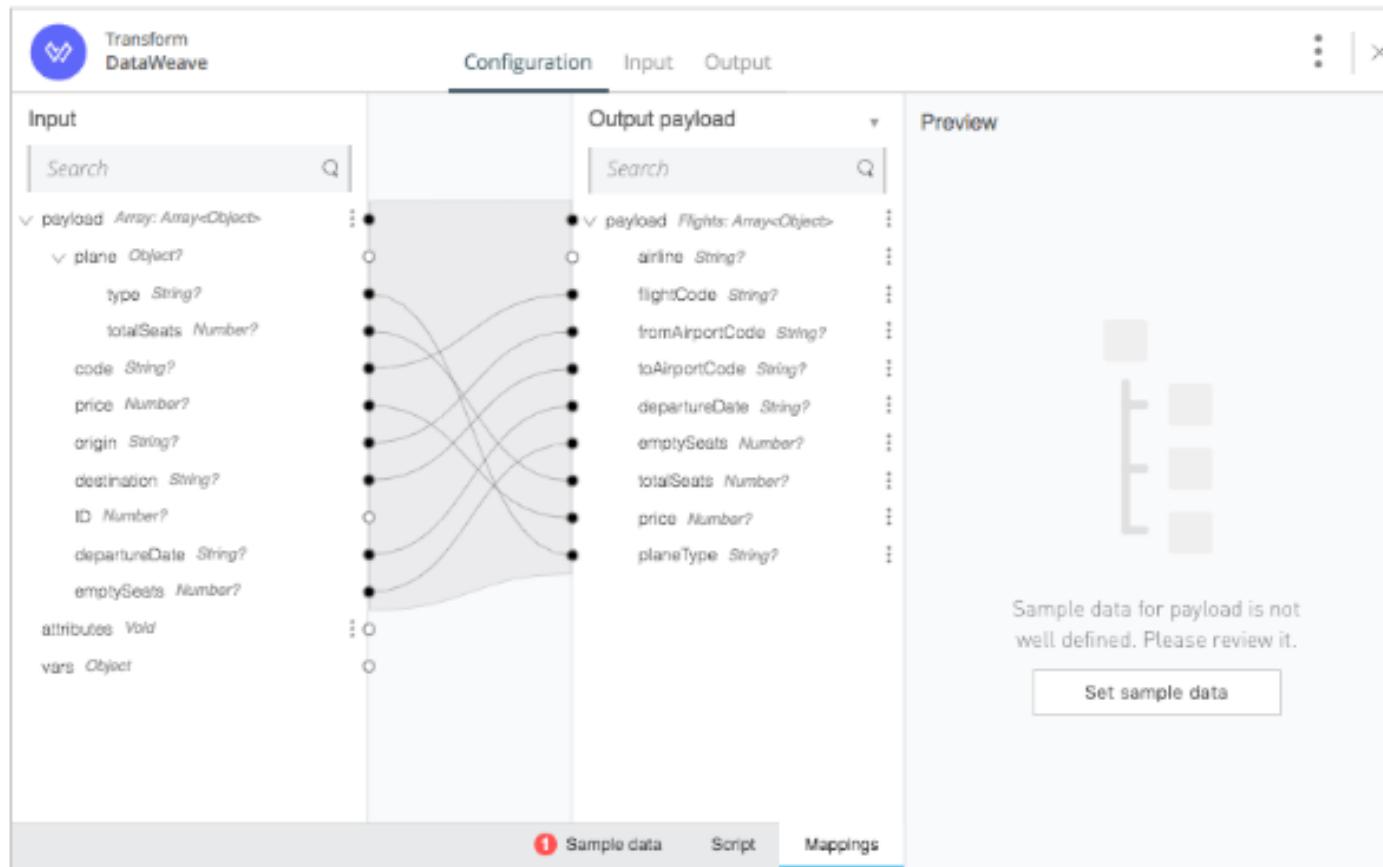
39. Map fields with the same names by dragging them from the input section and dropping them on the corresponding field in the output section.

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



40. 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



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

Output payload

Search

payload Flights: Array<Object>

- airline String?
- flightCode String?
- fromAirportCode String?
- toAirportCode String?

Preview

Data type actions

- Create
- Edit
- Set
- Detach
- Set Expression

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

payload Flights: Array<Object>

airline String?

"american"

Cancel Ok

43. 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 transformations later in the Development Fundamentals course.

Transform
DataWeave

Configuration Input Output

Input

Search

payload Array:Array<Objects>

 └ plane Object?

 type String?

 totalSeats Number?

 code String?

 price Number?

 origin String?

 destination String?

 ID Number?

 departureDate String?

 emptySeats Number?

 └ attributes Void

 └ vars Object

Transformation script

```
1  dw 2.0
2  output application/json
3  ---
4  (payload map (value@, index@) -> {
5    flightCode: value@.code,
6    fromAirportCode: value@.origin,
7    toAirportCode: value@.destination,
8    departureDate: value@.departureDate,
9    emptySeats: value@.emptySeats,
10   totalSeats: value@.plane.totalSeats,
11   price: value@.price,
12   planeType: value@.plane."type",
13   airline: "american"
14 })
```

Preview

Sample data for payload is not well defined. Please review it.

Set sample data

Sample data

Script

Mappings

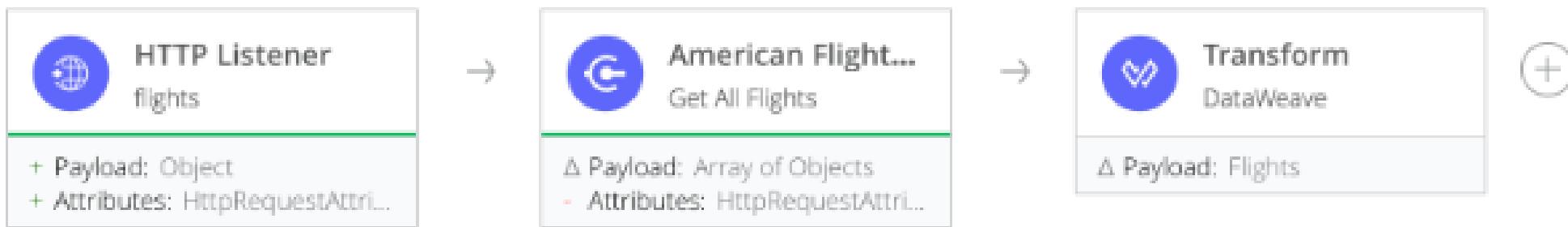
Add sample data

44. Click the Set sample data button in the preview section.
45. In the computer's file explorer, return to the student files folder and locate the american-flights-example.json file in the examples folder.
46. Open the file in a text editor and copy the code.
47. Return to flow designer and paste the code in the sample data for payload section.

The screenshot shows the DataWeave Transform tool interface. The top navigation bar includes 'Transform', 'DataWeave', 'Configuration' (which is selected), 'Input', and 'Output'. Below the navigation is a search bar labeled 'Search' and a dropdown menu for 'payload' (Array<Object>). The 'Input' panel on the left contains JSON code for defining the flight data structure. The 'Sample data for payload (application/json)' panel in the center contains two flight objects. The 'Preview' panel on the right shows the same two flight objects with line numbers 1 through 24. The bottom navigation bar has tabs for 'Sample data' (which is selected), 'Script', and 'Mappings'.

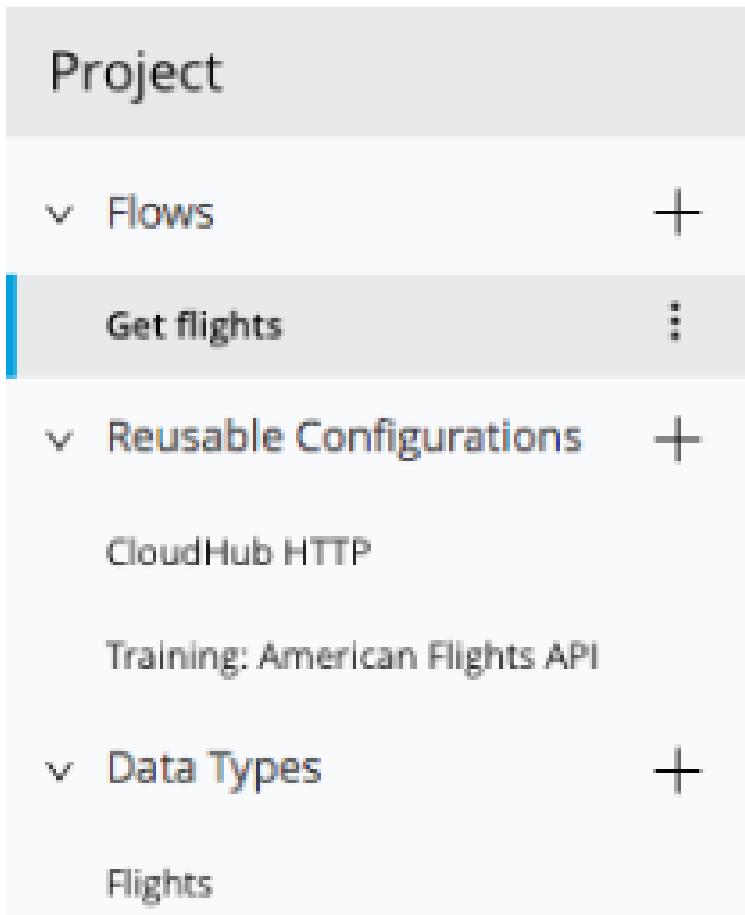
```
1  [
2    {
3      "ID": 1,
4      "code": "ER38sd",
5      "price": 400.00,
6      "departureDate": "2016/03/20",
7      "origin": "MUA",
8      "destination": "SFO",
9      "emptySeats": 0,
10     "plane": {
11       "type": "Boeing 737",
12       "totalSeats": 150
13     },
14     "ID": 2,
15     "code": "ER451f",
16     "price": 345.99,
17     "departureDate": "2016/02/11",
18     "origin": "MUA",
19     "destination": "LAX",
20     "emptySeats": 52,
21     "plane": {
22       "type": "Boeing 777",
23       "totalSeats": 300
24   }
```

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



Locate the data type and configuration definitions

50. Locate the connector configuration and the new Flights data type in the project explorer.



Test the application

51. Run the project.
52. Return to Advanced REST Client 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.

The screenshot shows the Advanced REST Client interface. At the top, it displays the method (GET) and URL (http://americanflightsapp-dgzb.cloudhub.io/flights). Below the URL is a blue 'SEND' button and a vertical ellipsis menu. Underneath the URL, there's a 'Parameters' dropdown and a green '200 OK' status box indicating a successful response with a duration of 2087.69 ms. To the right of the status box is a 'DETAILS' dropdown. Below these, there are several icons: a copy icon, a refresh icon, a compare icon, a list icon, and a search icon. The main content area shows a JSON array of flight data. The first flight in the array has a detailed view shown below it, containing fields like flightCode, fromAirportCode, toAirportCode, departureDate, emptySeats, totalSeats, price, planeType, and airline. The second flight in the array is partially visible.

```
[{"flightCode": "rree0001", "fromAirportCode": "MUA", "toAirportCode": "LAX", "departureDate": "2016-01-20T00:00:00", "emptySeats": 0, "totalSeats": 200, "price": 541, "planeType": "Boeing 787", "airline": "american"}, {"flightCode": "eefd0123", "fromAirportCode": "MUA", "toAirportCode": "JFK", "departureDate": "2016-01-21T00:00:00", "emptySeats": 0, "totalSeats": 200, "price": 541, "planeType": "Boeing 787", "airline": "american"}]
```

Stop the application

53. Return to Runtime Manager.
54. In the left-side navigation, click Applications.
55. Select the row with your application; you should see information about the application displayed on the right side of the window.

The screenshot shows the Oracle Cloud Infrastructure Runtime Manager interface. On the left, there's a sidebar with 'DESIGN' selected, followed by 'Applications' (which is also selected), 'Servers', 'Alerts', 'VPCs', and 'Load Balancers'. The main area has tabs for 'Deploy application' and 'Search Applications'. A table lists applications with columns: Name, Server, Status, and File. One row is selected for 'americanflightsapp-dgzb' with 'CloudHub' as the server, 'Started' status, and 'americanflightsapp-dgzb.jar' as the file. To the right, a detailed view for 'americanflightsapp-dgzb' is shown. It includes a status bar ('Started' with a green dot, 'CloudHub'), a file section ('americanflightsapp-dgzb.jar' with a 'Choose file' button), deployment info ('Last Updated: 2018-03-25 10:58:12AM', 'App url: americanflightsapp-dgzb.cloudhub.us'), runtime details ('Runtime version: 4.1.0', 'Worker size: 0.2 vCores', 'Workers: 1'), and buttons for 'Manage Application', 'Logs', and 'Insight'. At the bottom, there's a link 'View Associated Alerts'.

56. 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.

The screenshot shows the Runtime Manager interface. On the left, there's a sidebar with tabs for DESIGN, Applications, Servers, Alerts, VPCs, and Load Balancers. The DESIGN tab is selected. In the center, there's a table with columns: Name, Server, Status, and File. A single row is selected, showing 'americanflightsapp-dgvb' as the name, 'CloudHub' as the server, 'Undeployed' as the status, and 'americanflightsapp-dgvb.jar' as the file. To the right of the table, a detailed view of the application is shown in a modal window. The modal has a header 'americanflightsapp-dgvb'. It shows the status as 'Undeployed' with a dropdown arrow, and 'CloudHub' selected. Below that is a file input field containing 'americanflightsapp-dgvb.jar' with a 'Choose file' button. Underneath, it says 'Last Updated: 2018-03-25 11:41:41AM' and 'App url: americanflightsapp-dgvb.cloudhub.io'. At the bottom of the modal, there are buttons for 'Manage Application', 'Logs', and 'Insight', and a link 'View Associated Alerts'.

57. Close Runtime Manager.

58. Return to flow designer; you should see the application is not running.

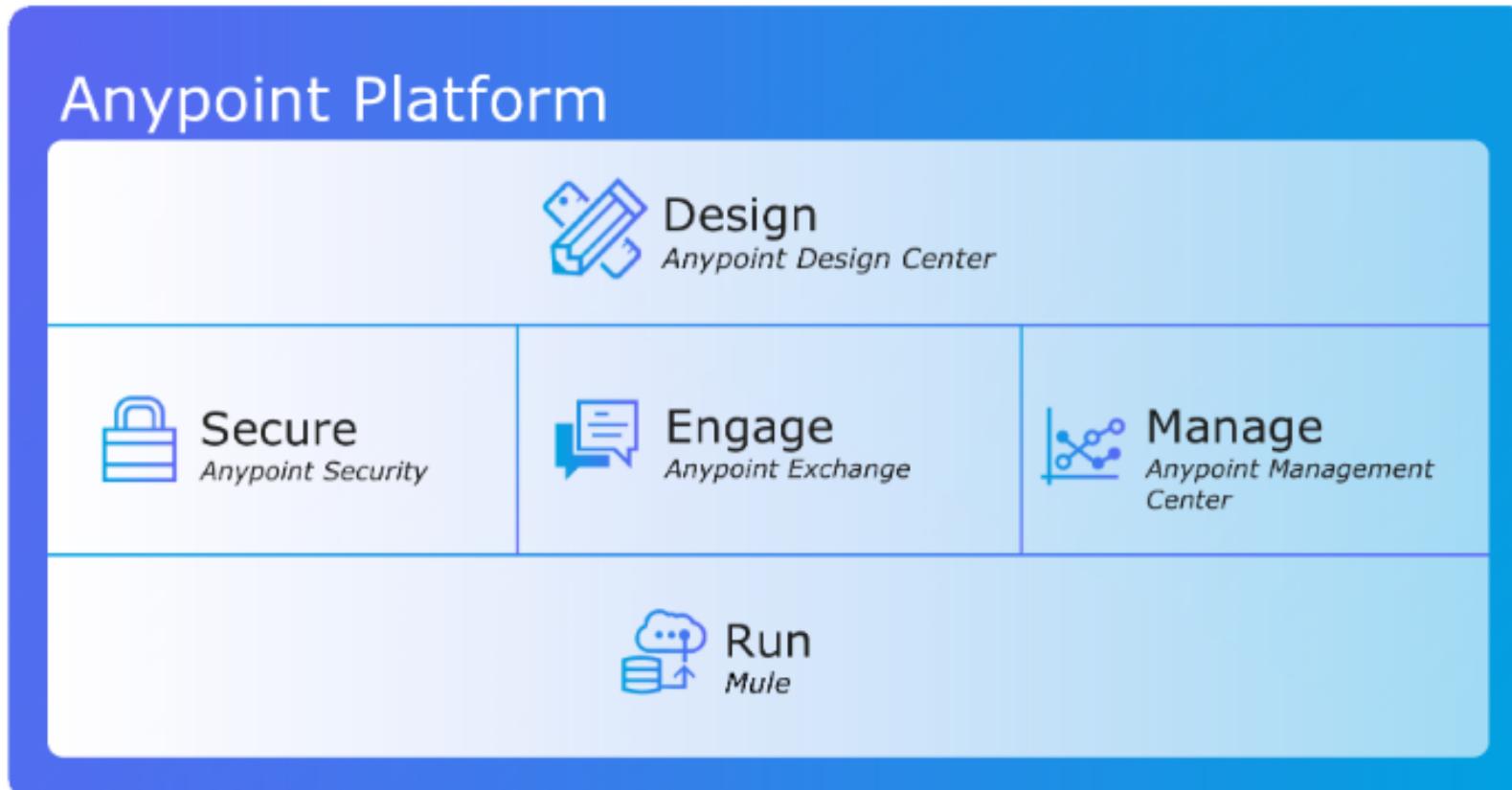
The screenshot shows the Design Center interface. At the top, it says 'Design Center > American Flights App'. Below that is a toolbar with icons for back, forward, search, and save, followed by the text 'Saved 7 minutes ago'. Then there's a green 'Run' button with the text 'Not running' next to it. To the right of the run button is a vertical ellipsis button, and further right is a blue 'Deploy' button.

59. Return to Design Center.

Summary



- **Anypoint Platform** is a unified, hybrid integration platform that creates a seamless **application network** of apps, data, and devices with **API-led connectivity**



- Use **Anypoint Exchange** as a central repository for assets so they can be discovered and reused
 - Populate it with everything you need to build your integration projects
- Use **flow designer** to build integration applications
 - These are Mule 4 applications that are deployed to a Mule runtime
 - To learn more, take the 1-day *Anypoint Platform: Flow Design* course
- **Mule runtimes** can be MuleSoft-hosted in the cloud (CloudHub) or customer-hosted in the cloud or on-prem
- **DataWeave 2.0** is the expression language for Mule to access, query, and transform Mule 4 event data