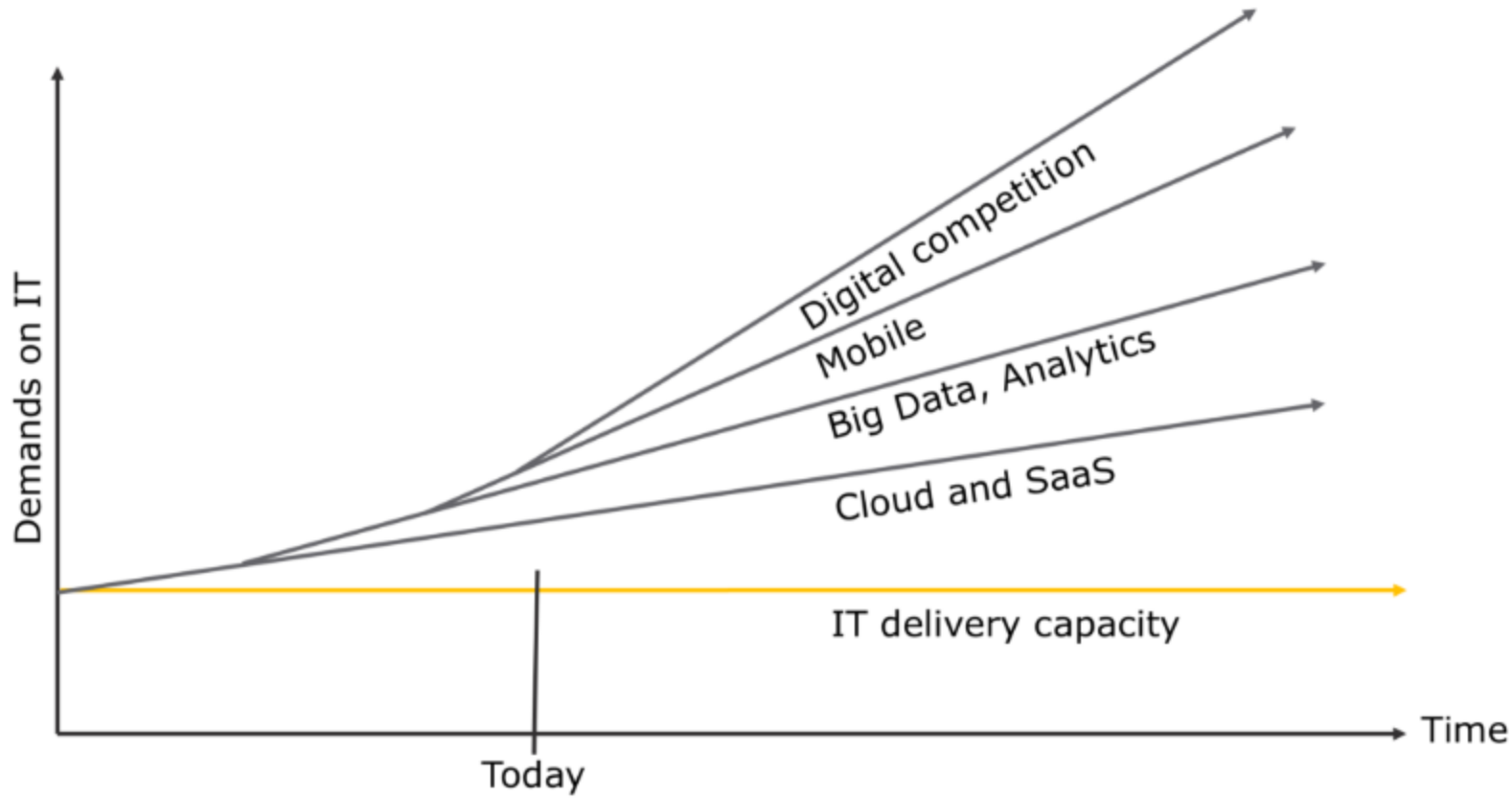
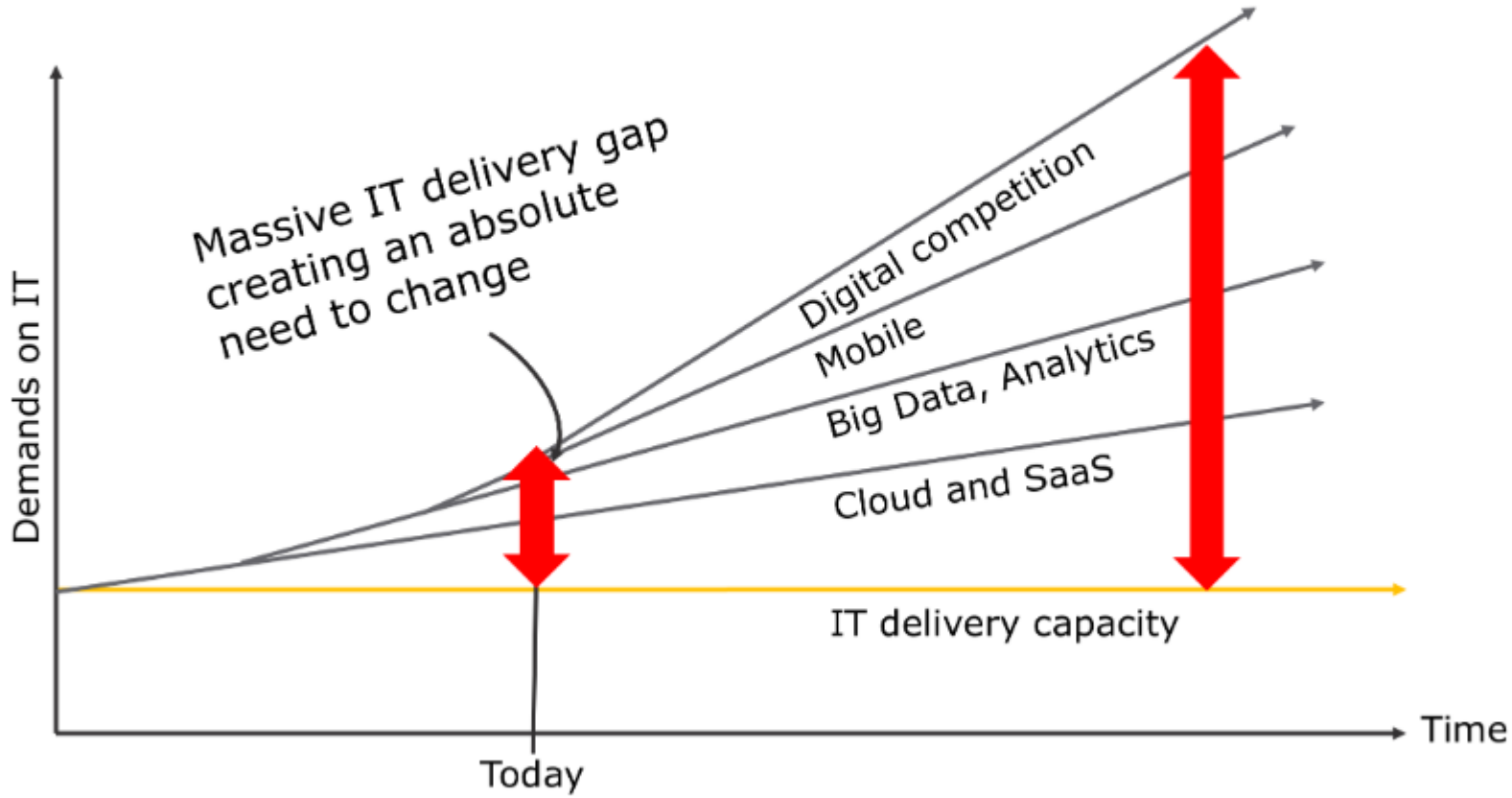
Identifying the problems faced by IT today

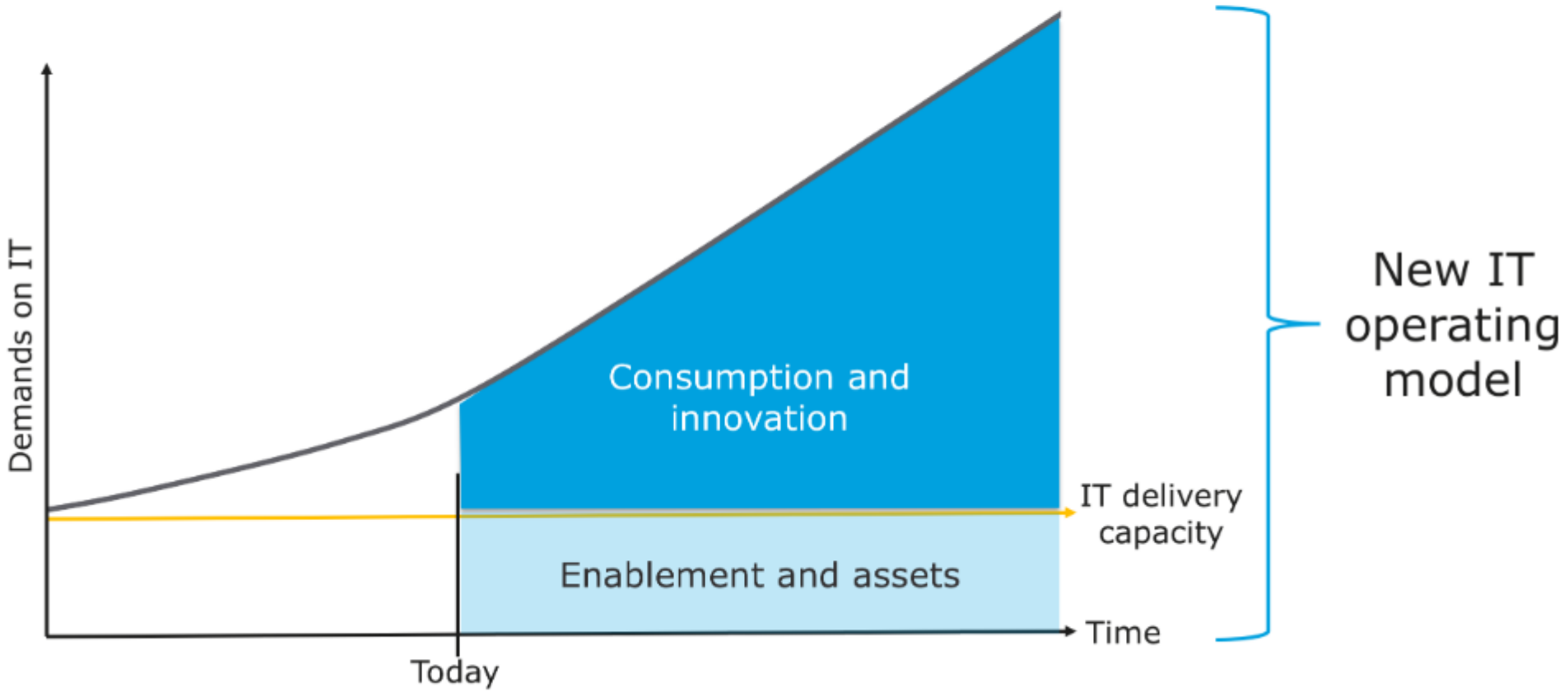
Biggest challenge: IT cannot go fast enough



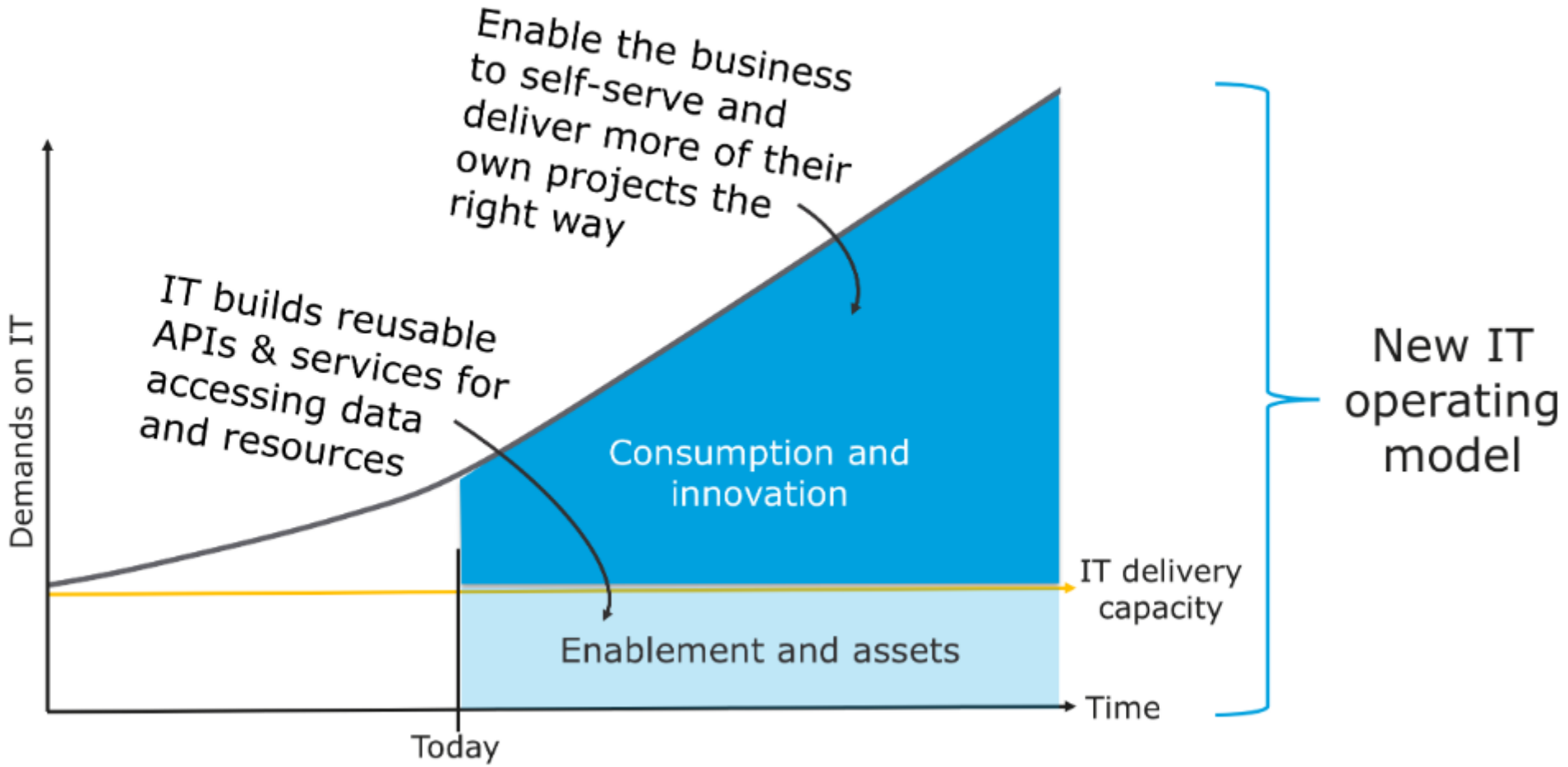
Digital pressures create a widening IT delivery gap



A new way of working to close the delivery gap

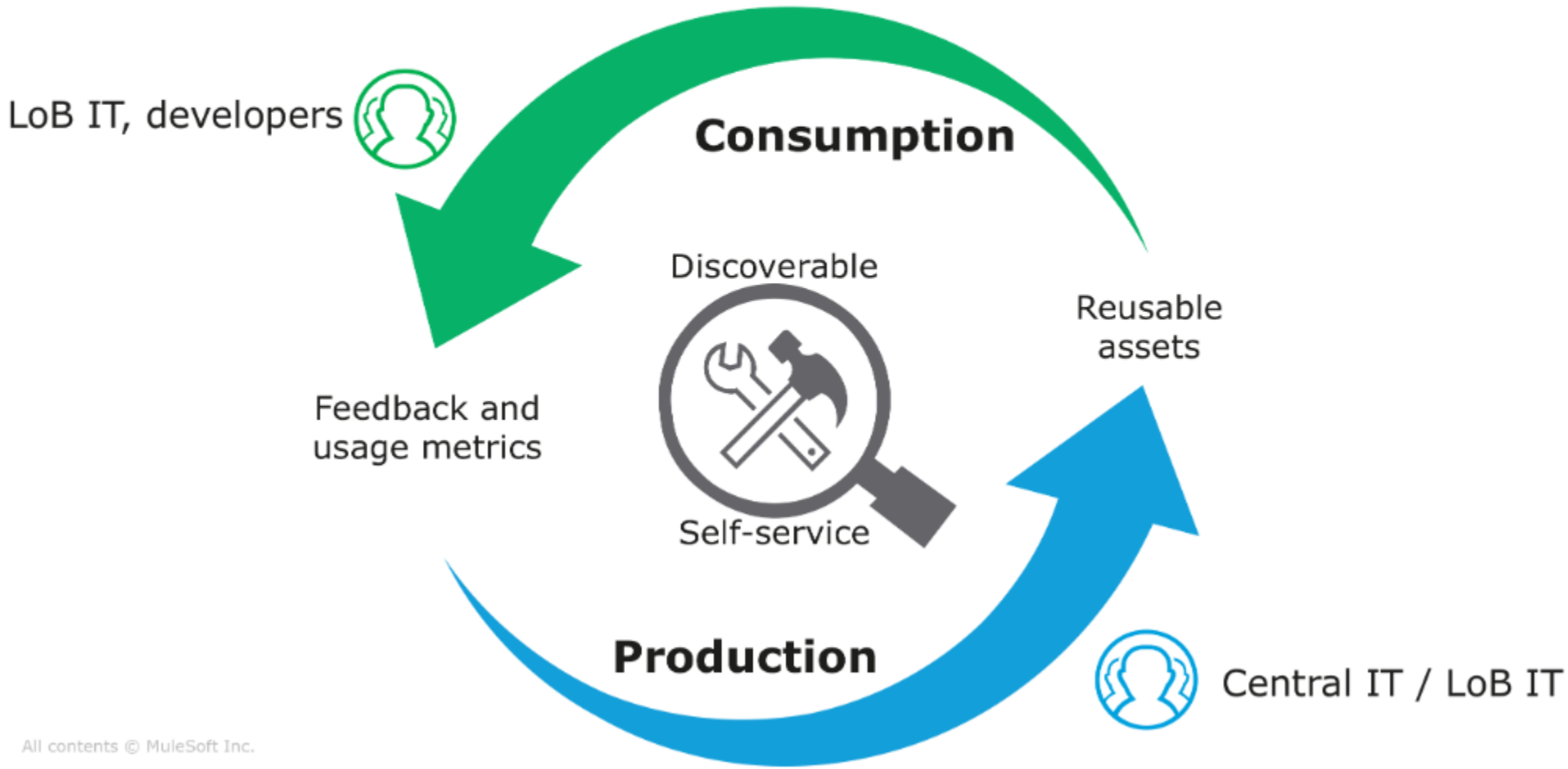


A new way of working to close the delivery gap



Introducing a new IT operating model

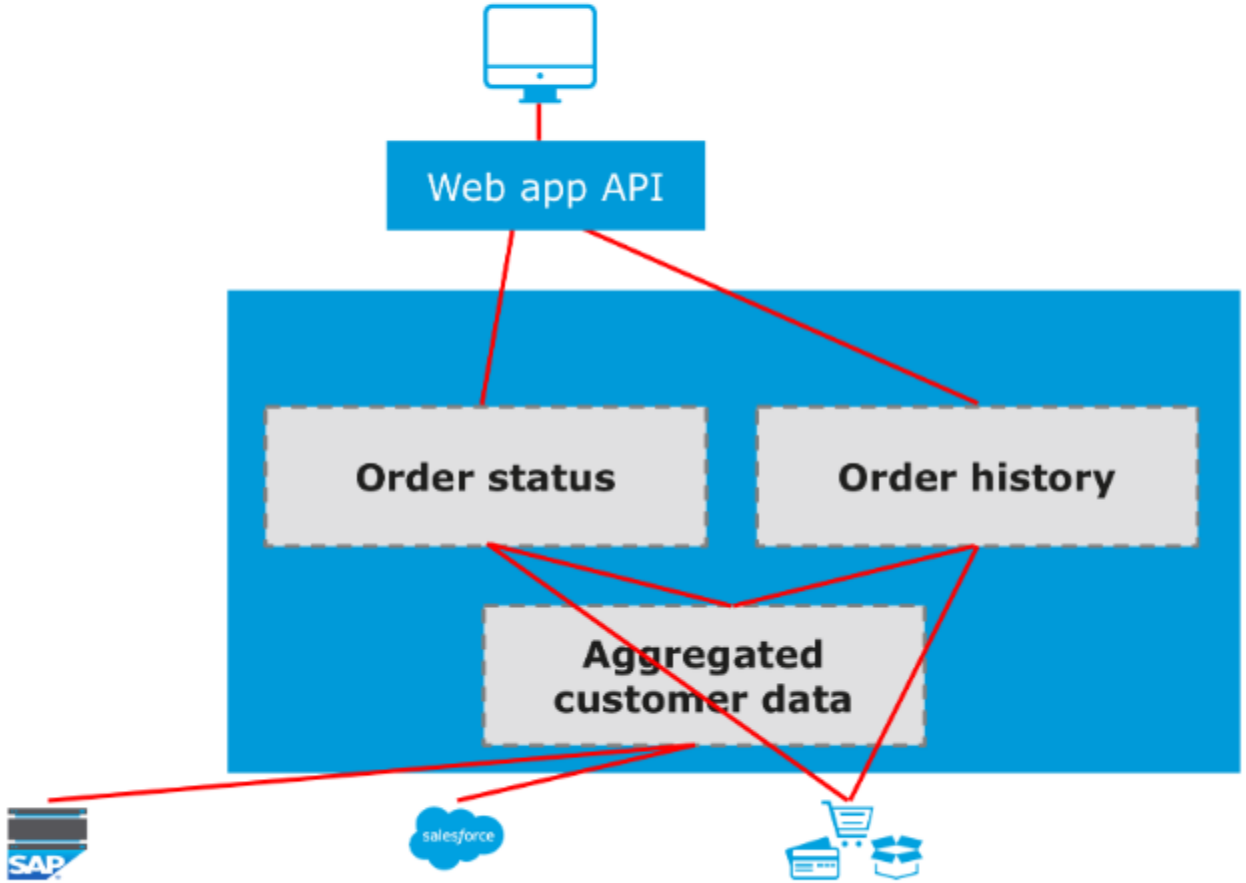
New operating model emphasizes consumption



A common project-based approach

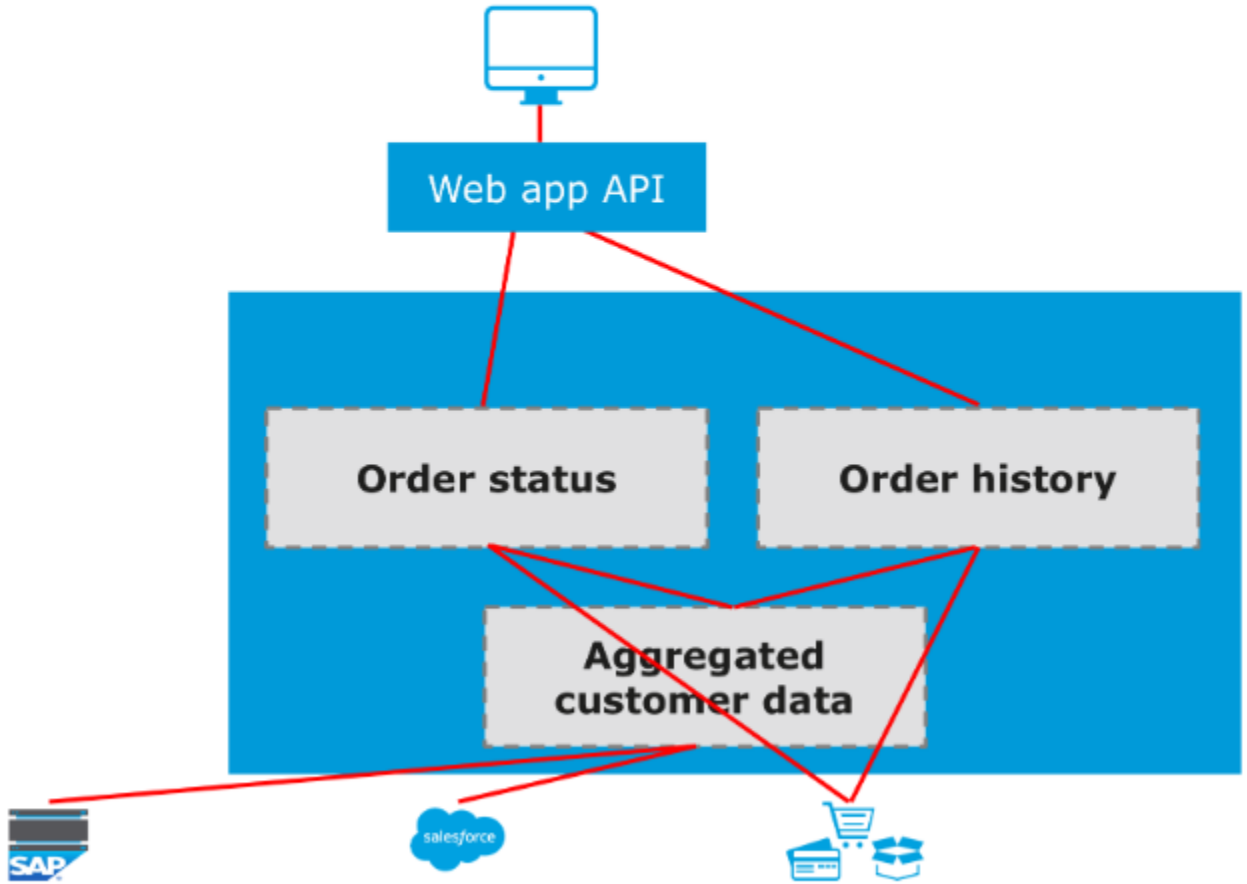
Project objective: Web app provides real-time order status and order history for sales team engaging with customers

- Order data in eCommerce system
- Inventory data in SAP
- Customer data in SAP, Salesforce

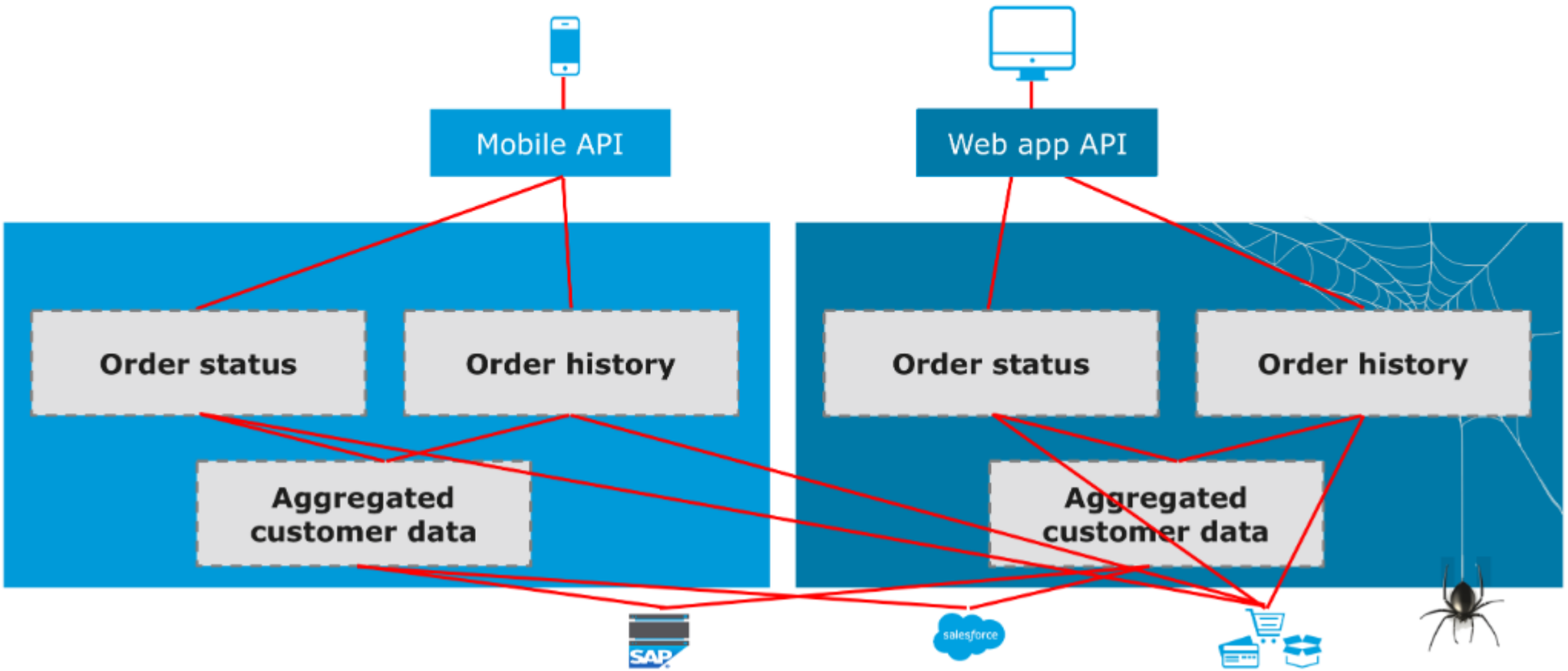


A common project-based approach

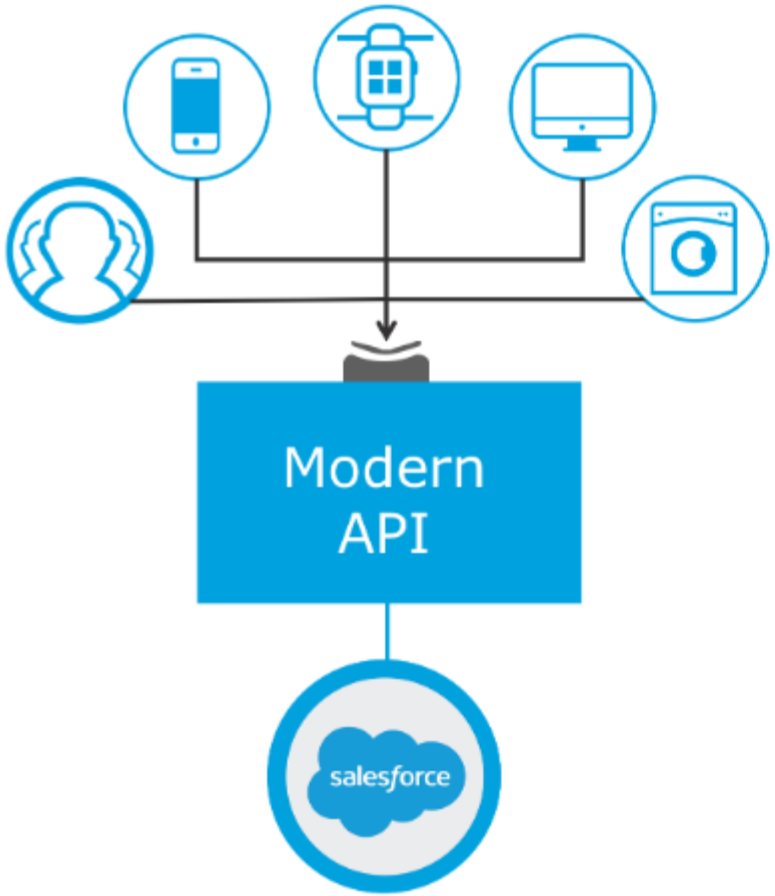
- On time and within budget
- Limited opportunity for reuse
- Tight coupling = brittleness
- Difficult to govern
- Meets business requirements



6 months later…

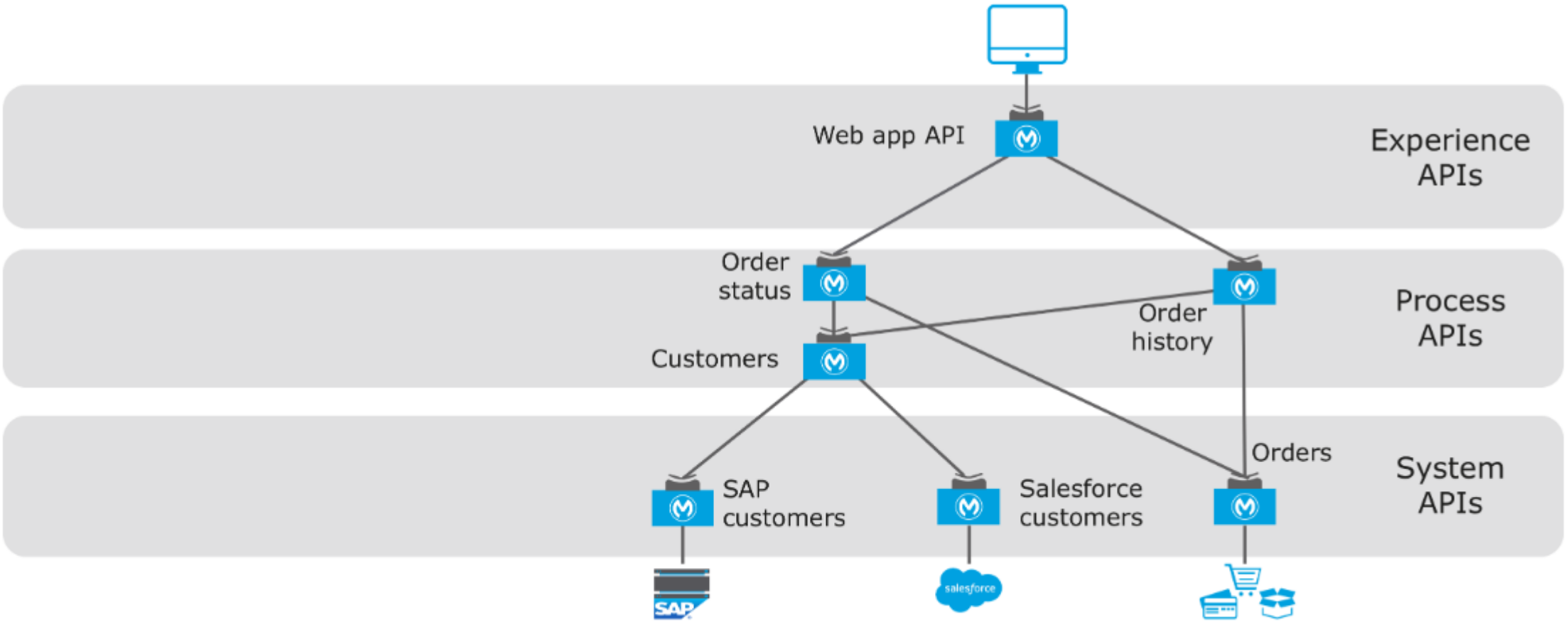


Modern API: The core enabler of a new operating model

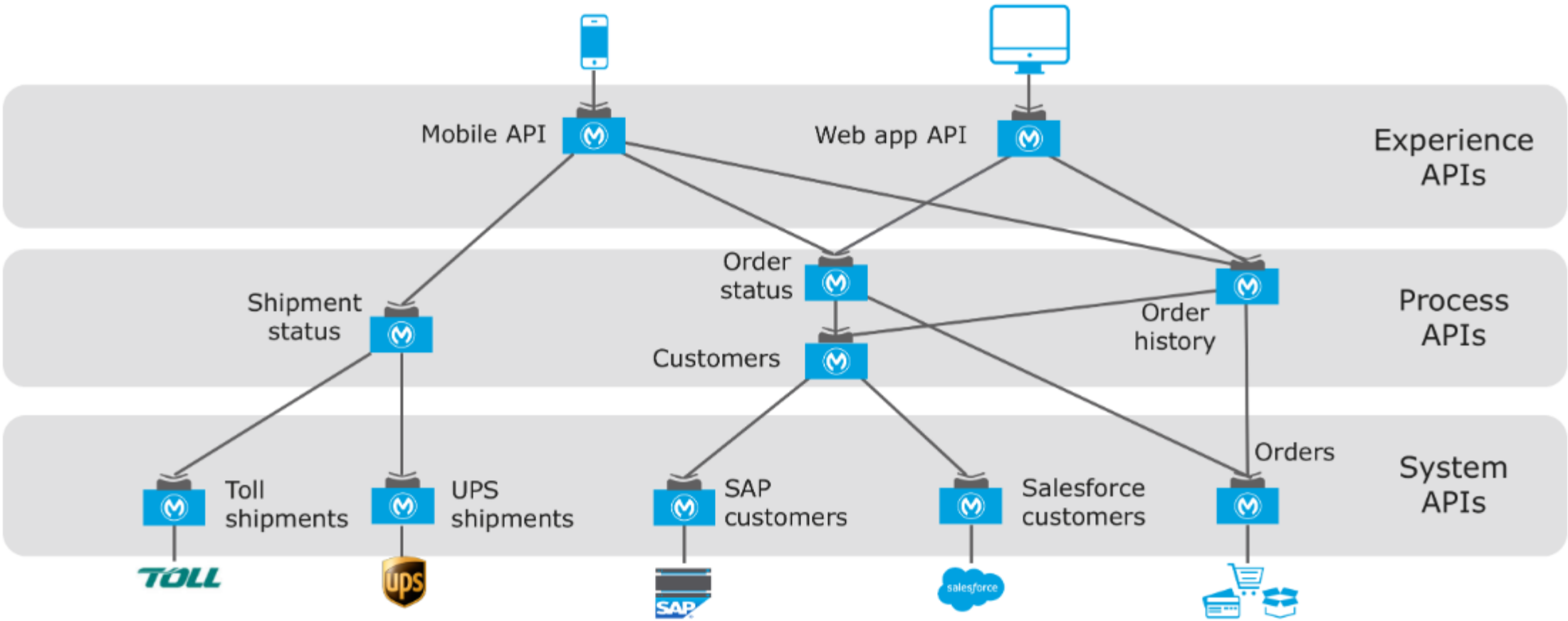


- Discoverable and accessible through self-service
- Productized and designed for ease of consumption
- Easily managed for security, scalability, and performance

The API-led connectivity approach



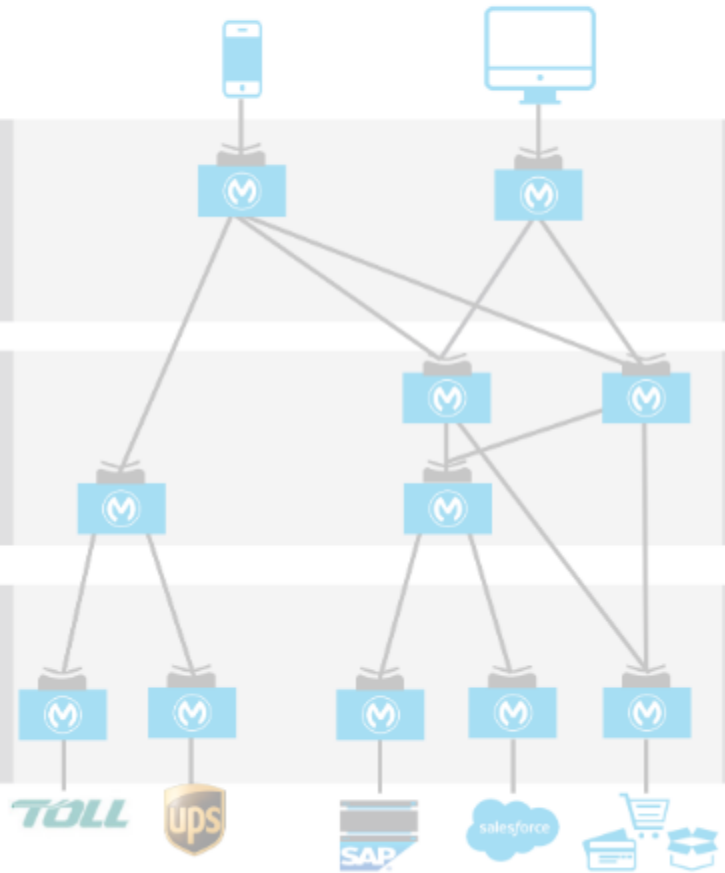
The API-led connectivity approach



Enable and empower the entire organization

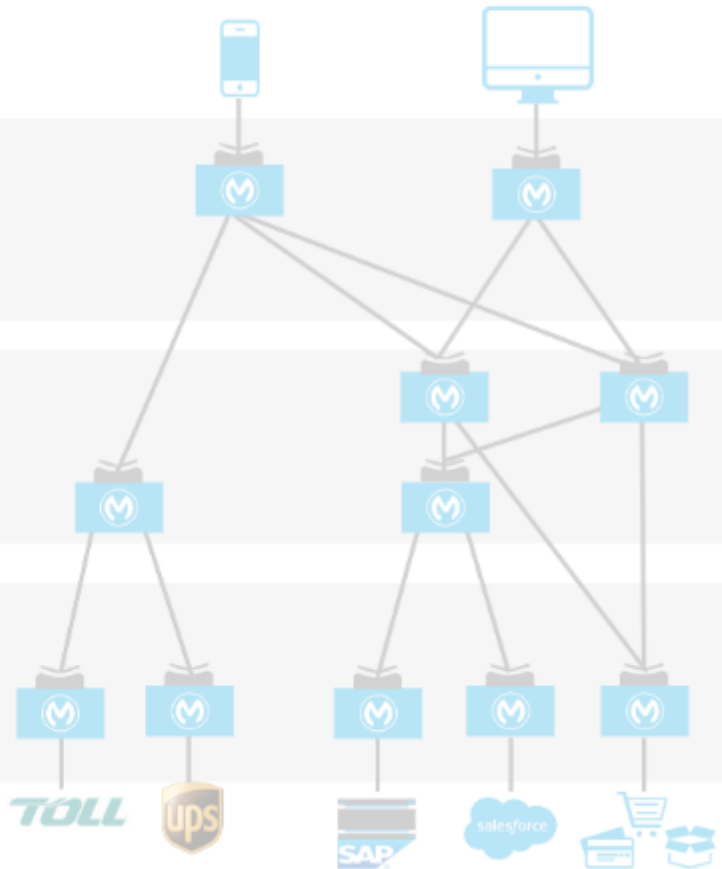
MuleSoft

Developers	Discover, self-serve, reuse and **consume**	Experience APIs
LOB IT	Discover, reuse assets and **compose** information	Process APIs
Central IT	**Unlock** assets and decentralize access	System APIs

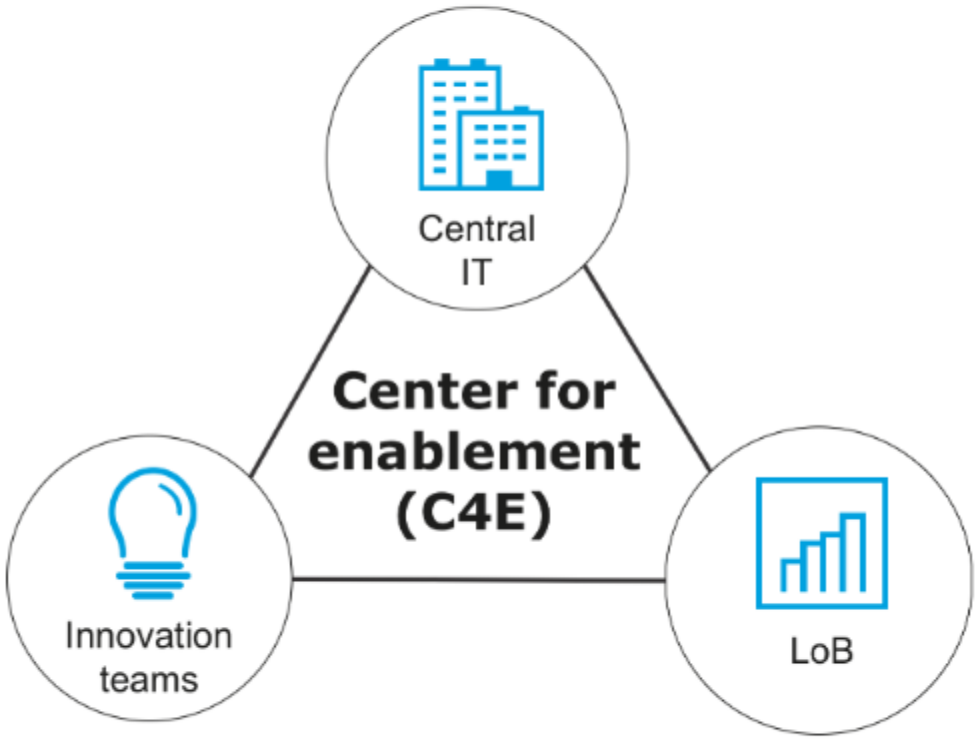


Drive outcomes with API-led connectivity

- On time and within budget
- Drives reuse vs build
- Designs in readiness for change
- Builds in governance, compliance, security, and scalability
- Meets the needs of your business



C4E: Organizing differently to drive API-led connectivity



- C4E is a cross functional team
- C4E ensures that assets are
  - Productized and published
  - Consumable
  - Consumed broadly
  - Fully leveraged
- Success of C4E measured on asset consumption

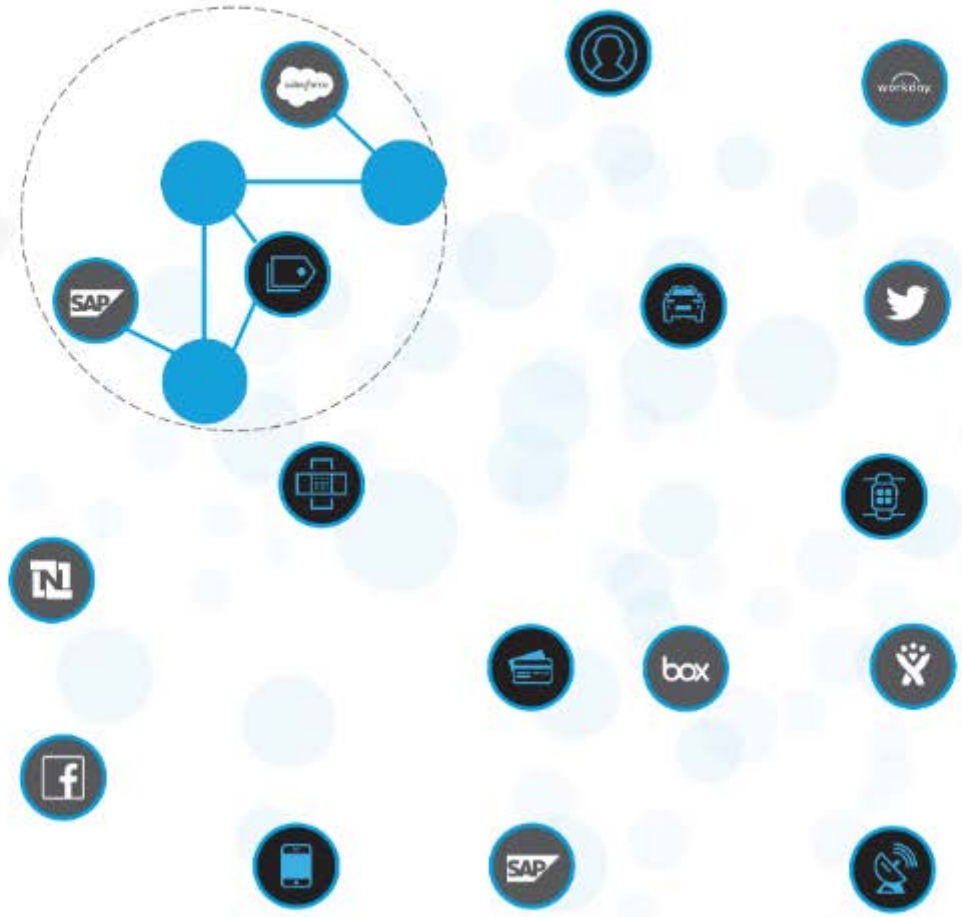
Achieving an application network

Application landscape

Every project adds value to the application network

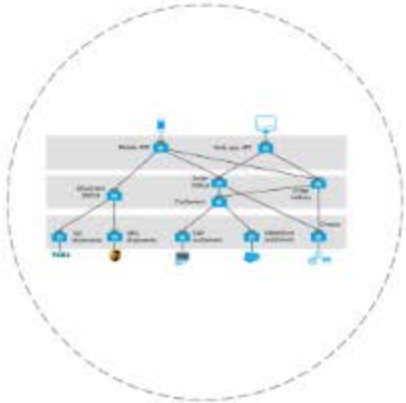
MuleSoft

Project 1 – API-led



Every project adds value to the application network

Project 1 – API-led



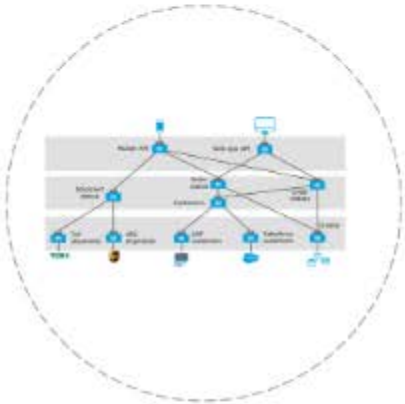
C4E

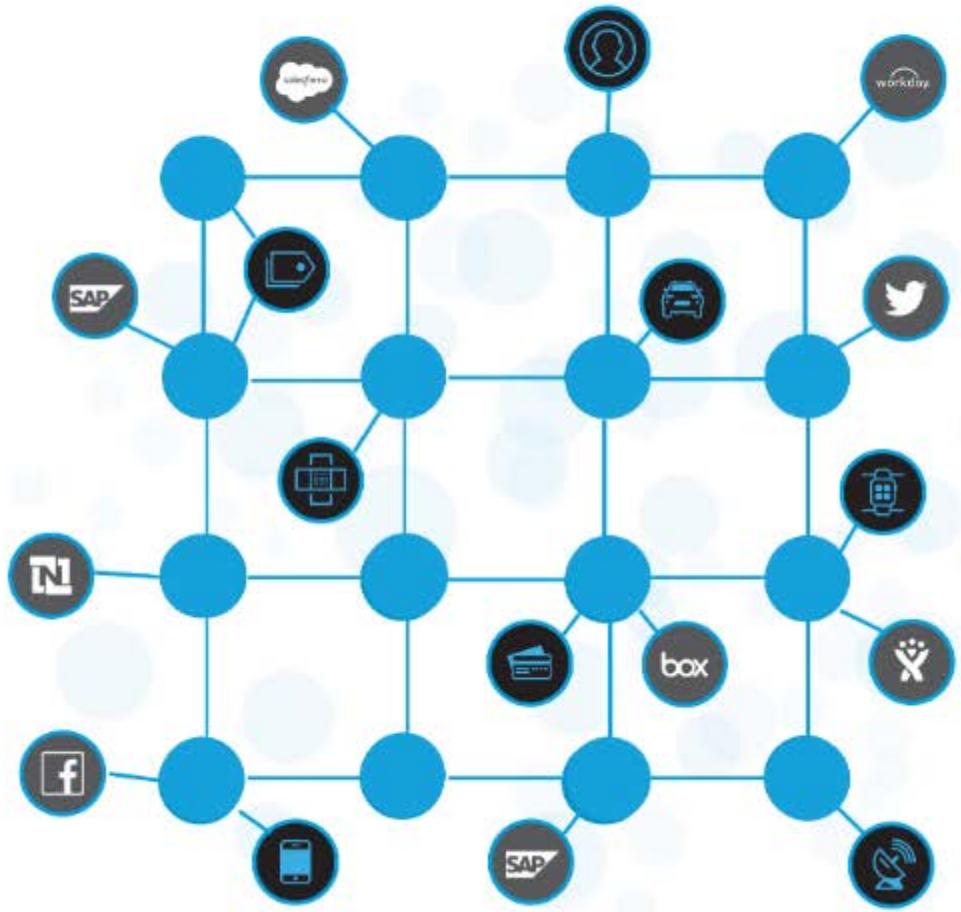
Self-serve assets on the application network

- Order status
- Shipments
- Orders
- Shipment status
- Customers

Every project adds value to the application network

Project 1 – API-led



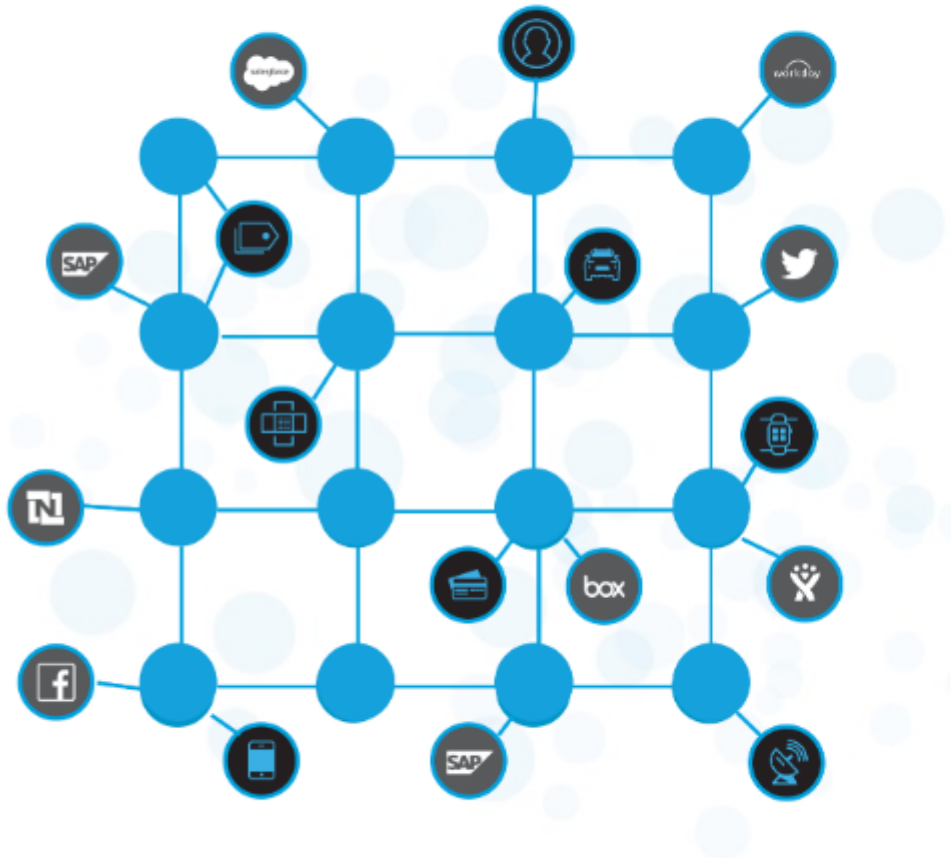


C4E

Self-serve assets on the application network

- Order status
- Shipments
- Orders
- Shipment status
- Customers

Speed. Agility. Innovation.



An application network

- Emerges bottoms-up via self-service
- Provides visibility, security and governability at every API node
- Is recomposable: it bends, not breaks – **built for change**

Deconstructing APIs

What exactly is an API?

- An **API** is an **A**pplication **P**rogramming **I**nterface

- It provides the info for **how to communicate with a software component**, defining the
  - Operations (what to call)
  - Inputs (what to send with a call)
  - Outputs (what you get back from a call)
  - Underlying data types

- It defines **functionalities independent of implementations**
  - You can change what's going on behind the scenes without changing how people call it

What do people mean when they say API?

MuleSoft

They could be referring to a number of things…

1. **An API interface definition file (API specification)**
   - Defines what you can call, what you send it, and what you get back

2. **A web service**
   - The actual API implementation you can make calls to or the interface of that API implementation

3. **An API proxy**
   - An application that controls access to a web service, restricting access and usage through the use of an API gateway

Reviewing web services

What is a web service?

- Different software systems often need to exchange data with each other
  - Bridging protocols, platforms, programming languages, and hardware architectures
- A **web service** is a method of communication that allows two software systems to exchange data over the internet
- Systems interact with the web service in a manner prescribed by some defined rules of communication
  - How one system can request data from another system, what parameters are required, the structure of the return data, and more

The parts of a web service

- **The web service API**
  - Describes how you interact with the web service
  - It may or may not (though it should!) be explicitly defined in a file
  - It could be any sort of text in any type of file but ideally should implement some standard API description language (or specification)
- **The web service interface implementing the API**
  - Is the code providing the structure to the application so it implements the API
  - This may be combined with the actual implementation code
- **The web service implementation itself**
  - Is the actual code and application

Two main types of web services

- SOAP web services
  - Traditional, more complex type
  - The communication rules are defined in an XML-based WSDL (Web Services Description Language) file

- **RESTful web services**
  - Recent, simpler type
  - Use the existing HTTP communication protocol

Reviewing RESTful web services

RESTful web services

- REST stands for **R**epresentational **S**tate **T**ransfer
  - An architectural style where clients and servers exchange representations of resources using standard HTTP protocol
- Other systems interact with the web service using the HTTP protocol
- The HTTP request method indicates which operation should be performed on the object identified by the URL

Example RESTful web service calls

- Data and resources are represented using URIs
- Resources are accessed or changed using a fixed set of operations

- (GET)/companies
- (GET)/companies?country=France
- (GET)/companies/3
- (POST)/companies with JSON/XML in HTTP body
- (DELETE)/companies/3
- (PUT)/companies/3 with JSON/XML in HTTP body

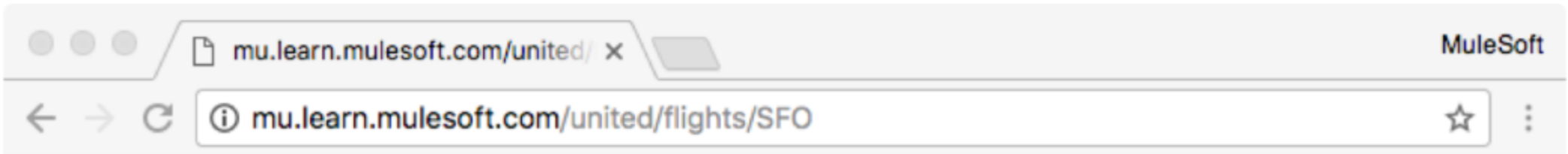
RESTful web service request methods

- **GET** retrieves the current state of a resource in some representation (usually JSON or XML)
- **POST** creates a new resource
- **DELETE** deletes a resource
- **PUT** replaces a resource completely
  - If the resource doesn't exist, a new one is created
- **PATCH** partially updates a resource
  - Just submitted data

Example RESTful web service response

MuleSoft

- JSON (JavaScript Object Notation)
  - A lightweight data-interchange format (without a lot of extra XML markup)
  - Human-readable results (usually JSON or XML)
  - Supports collections and maps



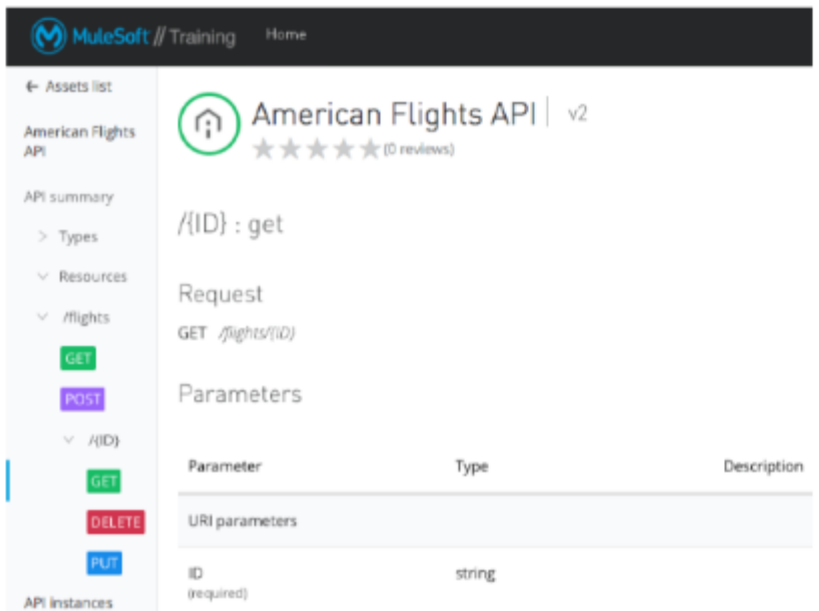

```
{"flights":
[{"code":"ER38sd","price":400,"origin":"MUA","destination":"SFO","departureDate":"2015/03/20",
"planeType":"Boeing 737","airlineName":"United","emptySeats":0},
{"code":"ER39rk","price":945,"origin":"MUA","destination":"SFO","departureDate":"2015/09/11","
planeType":"Boeing 757","airlineName":"United","emptySeats":54},
{"code":"ER39rj","price":954,"origin":"MUA","destination":"SFO","departureDate":"2015/02/12","
planeType":"Boeing 777","airlineName":"United","emptySeats":23}]}
```

Learning about APIs

- API documentation
  - Should include the list of all possible resources, how to get access to the API, and more
- API portals
  - Accelerate onboarding by providing developers a centralized place for discovering all the tools they need to successfully use the API, which could include
    - Documentation, tutorials, code snippets, and examples
    - A way to register applications to get access to the API
    - A way to provide feedback and make requests
    - A way to test the API by making calls to it
- Discover APIs in API directories and marketplaces
  - For example, **ProgrammableWeb**, which has over 19,000 APIs

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

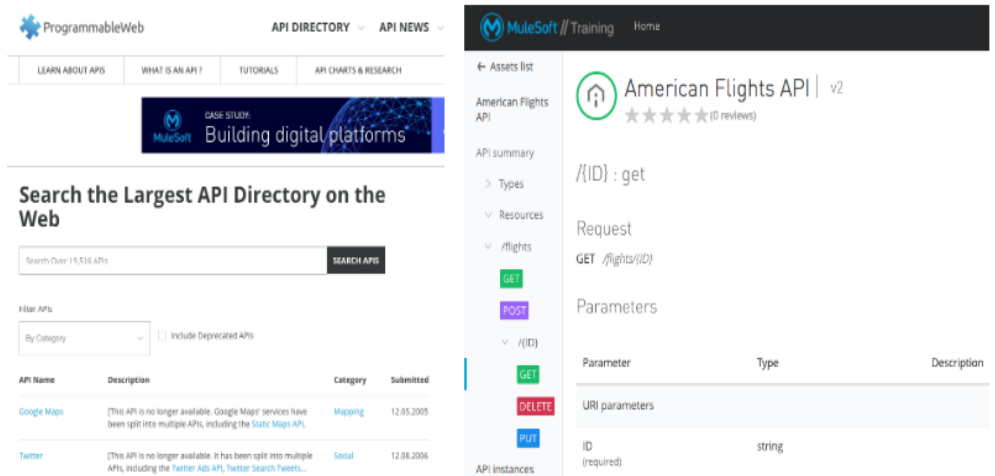
- Browse the ProgrammableWeb API directory
- Explore the API reference for an API (like Twitter)
- Explore the API portal for an API to be used in the course



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

In this walkthrough, you locate and explore documentation about APIs. You will:

- Browse the ProgrammableWeb API directory.
- Explore the API reference for an API (like Twitter).
- Explore the API portal for an API to be used in the course.

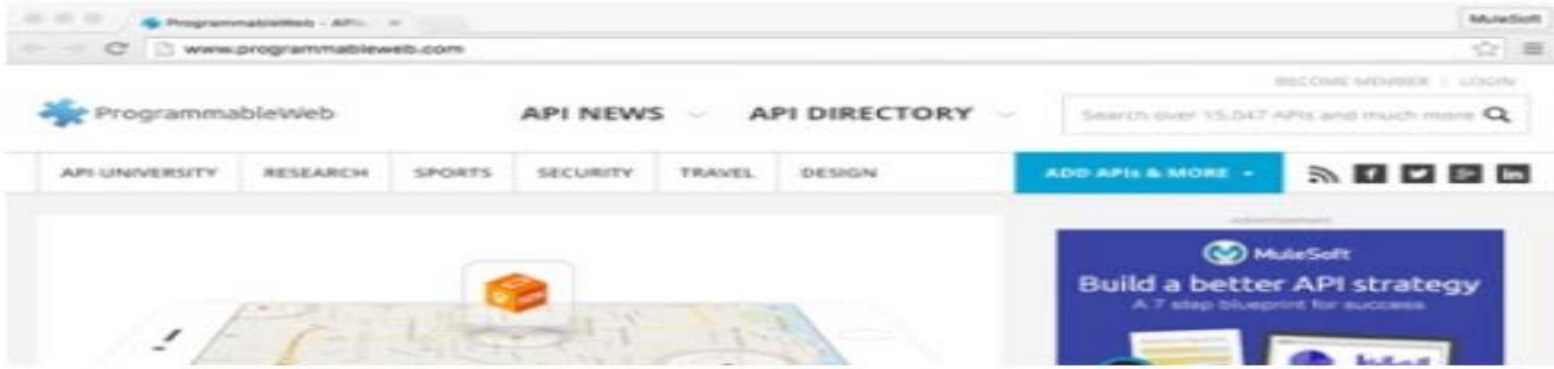


Browse the ProgrammableWeb API directory

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

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.

LEARN ABOUT APIS | WHAT IS AN API ? | API NEWS | API DIRECTORY

API Name	Description	Category	Submitted
Google Maps	[This API is no longer available. Google Maps' services have been split into multiple APIs, including the Static Maps API,	Mapping	12.05.2005
Twitter	[This API is no longer available. It has been split into multiple APIs, including the Twitter Ads API, Twitter Search Tweets...	Social	12.08.2006
YouTube	The Data API allows users to integrate their program with YouTube and allow it to perform many of the operations available on the website. It provides the capability to search for videos, retrieve...	Video	02.08.2006
Flickr	The Flickr API can be used to retrieve photos from the Flickr photo sharing service using a variety of feeds - public photos and videos, favorites, friends, group pools, discussions, and more. The...	Photos	09.04.2005

Explore the API reference for the Twitter API

4. Click the link for the Twitter API (or some other) API.
5. In the Specs section, click the API Portal / Home Page link.

TRACK API

Twitter API

Social Blogging

[This API is no longer available. It has been split into multiple APIs, including the Twitter Ads API, Twitter Search Tweets API, and Twitter Direct Message API.

This profile is maintained for historical, research, and reference purposes only.]

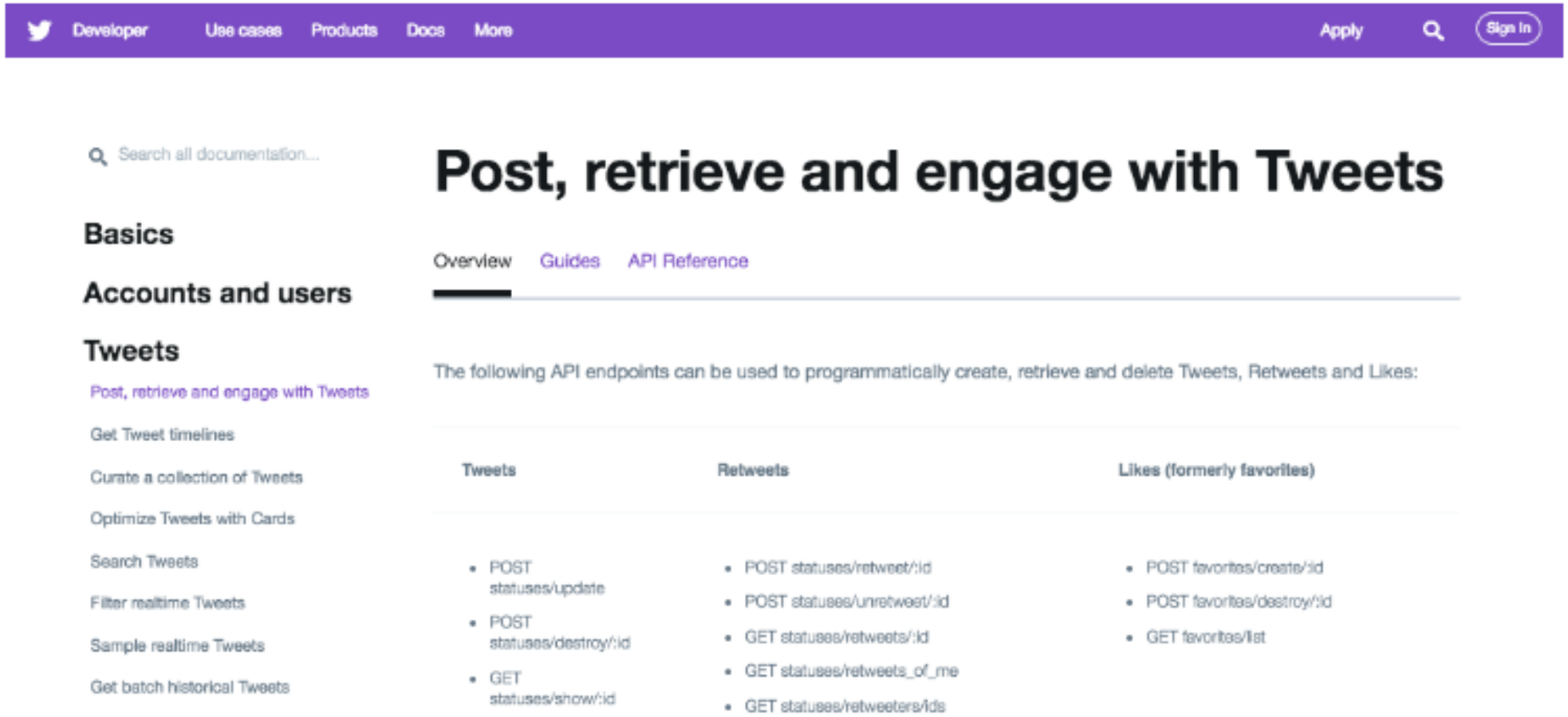
The Twitter micro-blogging service includes two RESTful APIs. The Twitter REST API methods allow developers to access core Twitter data. This includes update timelines, status data, and user information. The Search API methods give developers methods to interact with Twitter Search and trends data. The API presently supports the following data formats: XML, JSON, and the RSS and Atom syndication formats, with some methods only accepting a subset of these formats.

Summary	SDKs (72)	Articles (267)	How To (11)	Sample Source Code (25)	Libraries (38)	Developers (820)	Followers (1919)	Comments (4)

SPECS

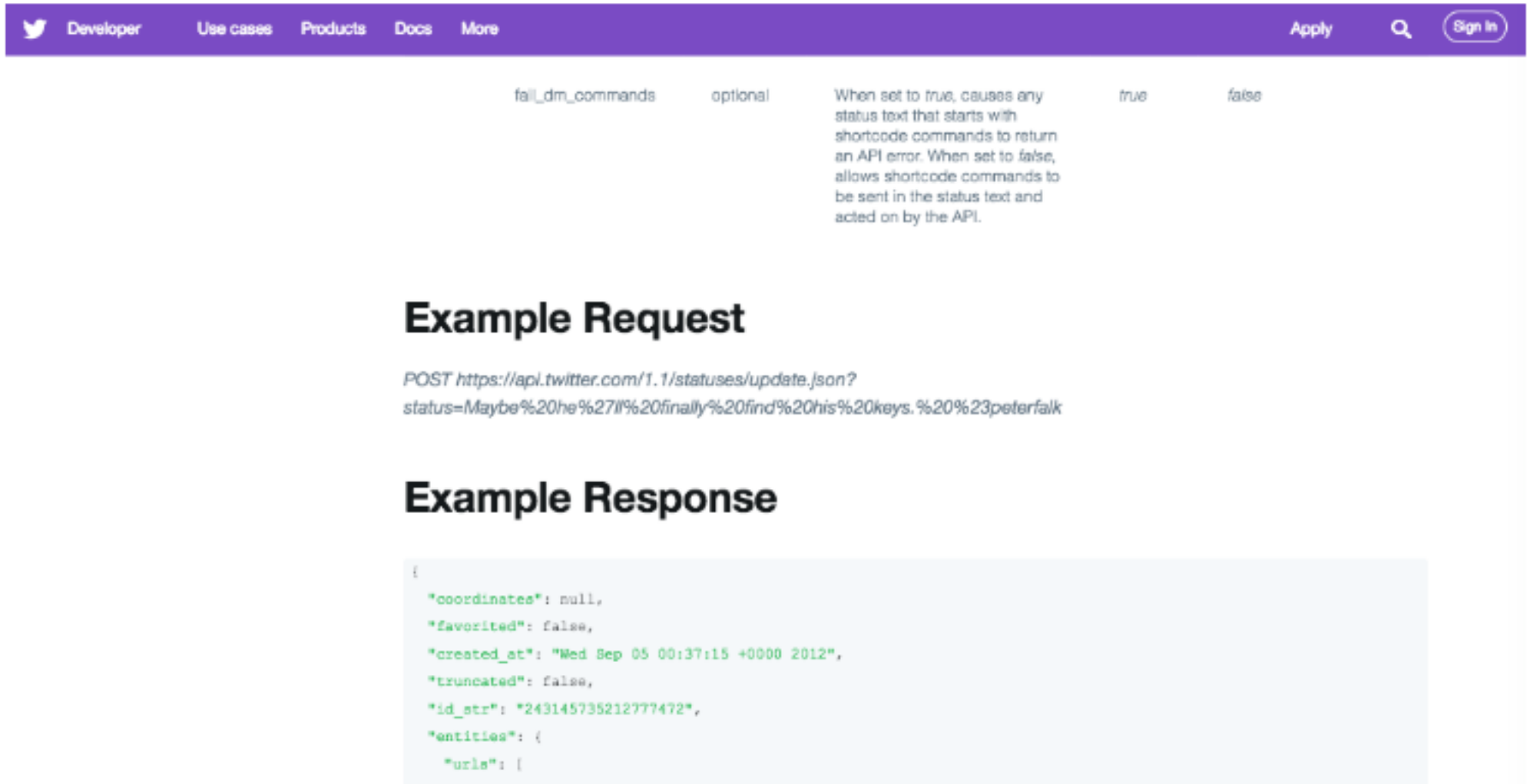
API Endpoint	http://twitter.com/statuses/
API Portal / Home Page	https://dev.twitter.com/rest/public
Primary Category	Social

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.

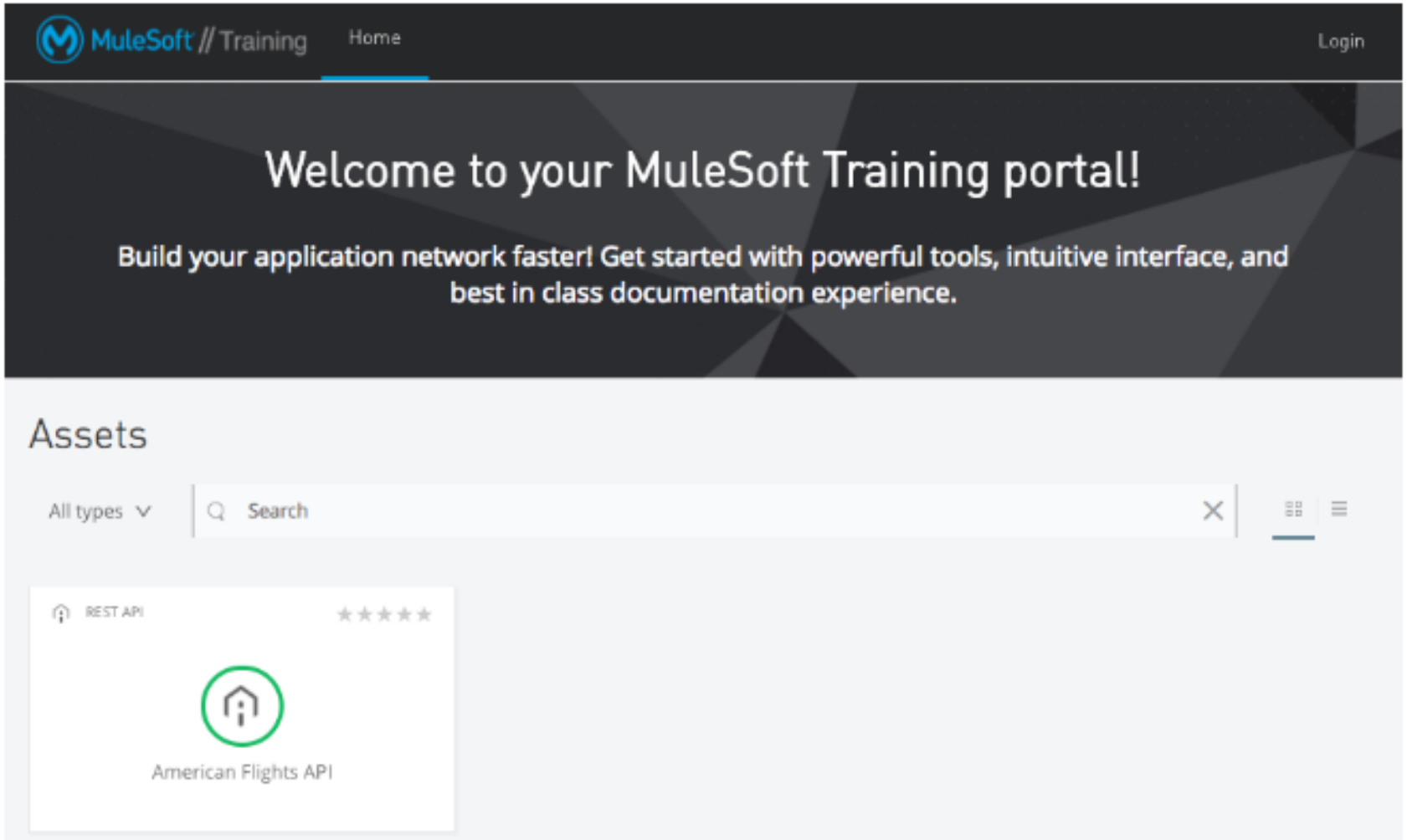
10. Review the information for POST statuses/update including parameters, example request, and example response.



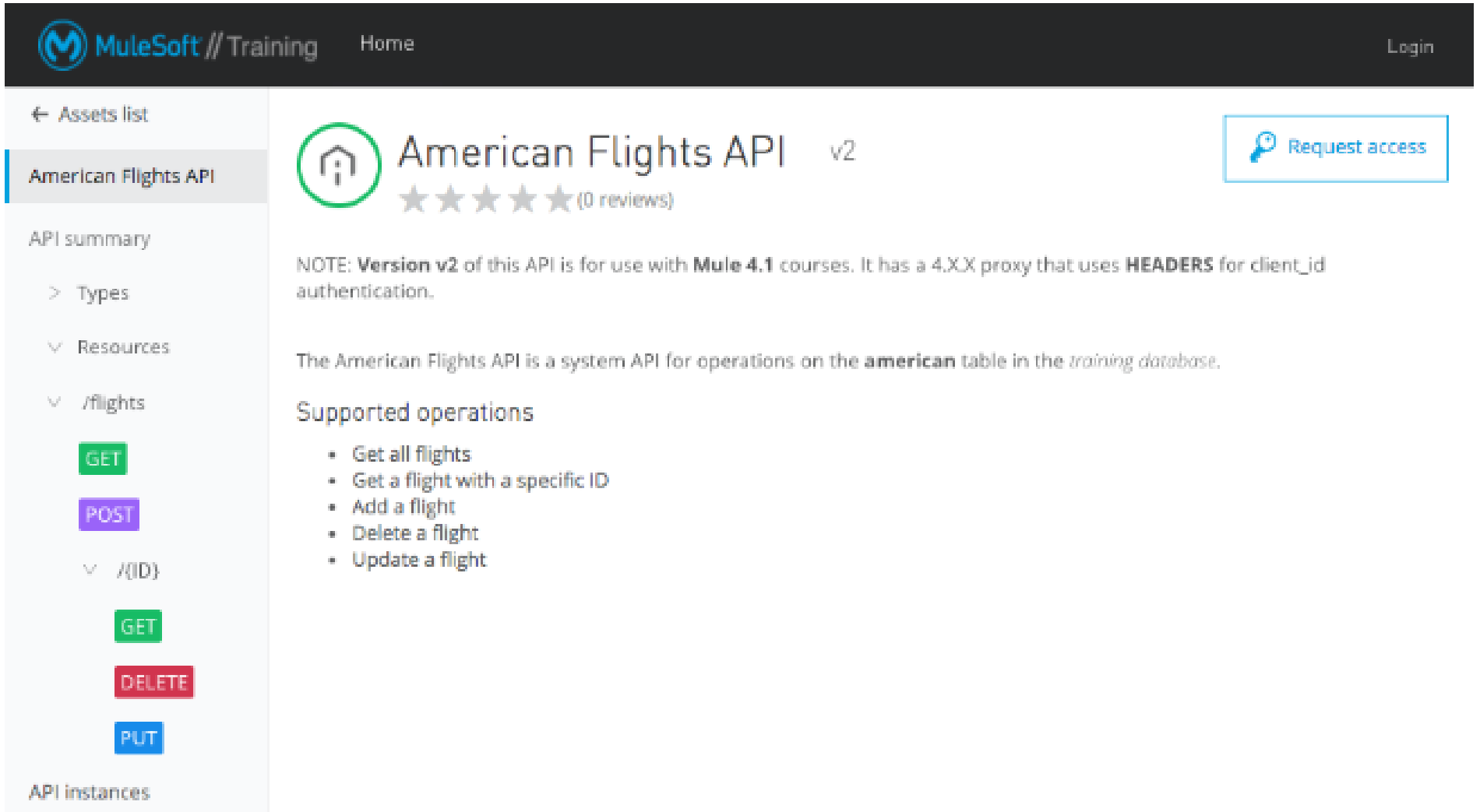
11. Close the browser tab.

Explore an API portal for an API to be used in the course

12. Return to the course snippets.txt file.
13. Copy the URL for the MuleSoft Training API portal.
14. Return to a browser window and navigate to that URL: https://anypoint.mulesoft.com/exchange/portals/muletraining/.

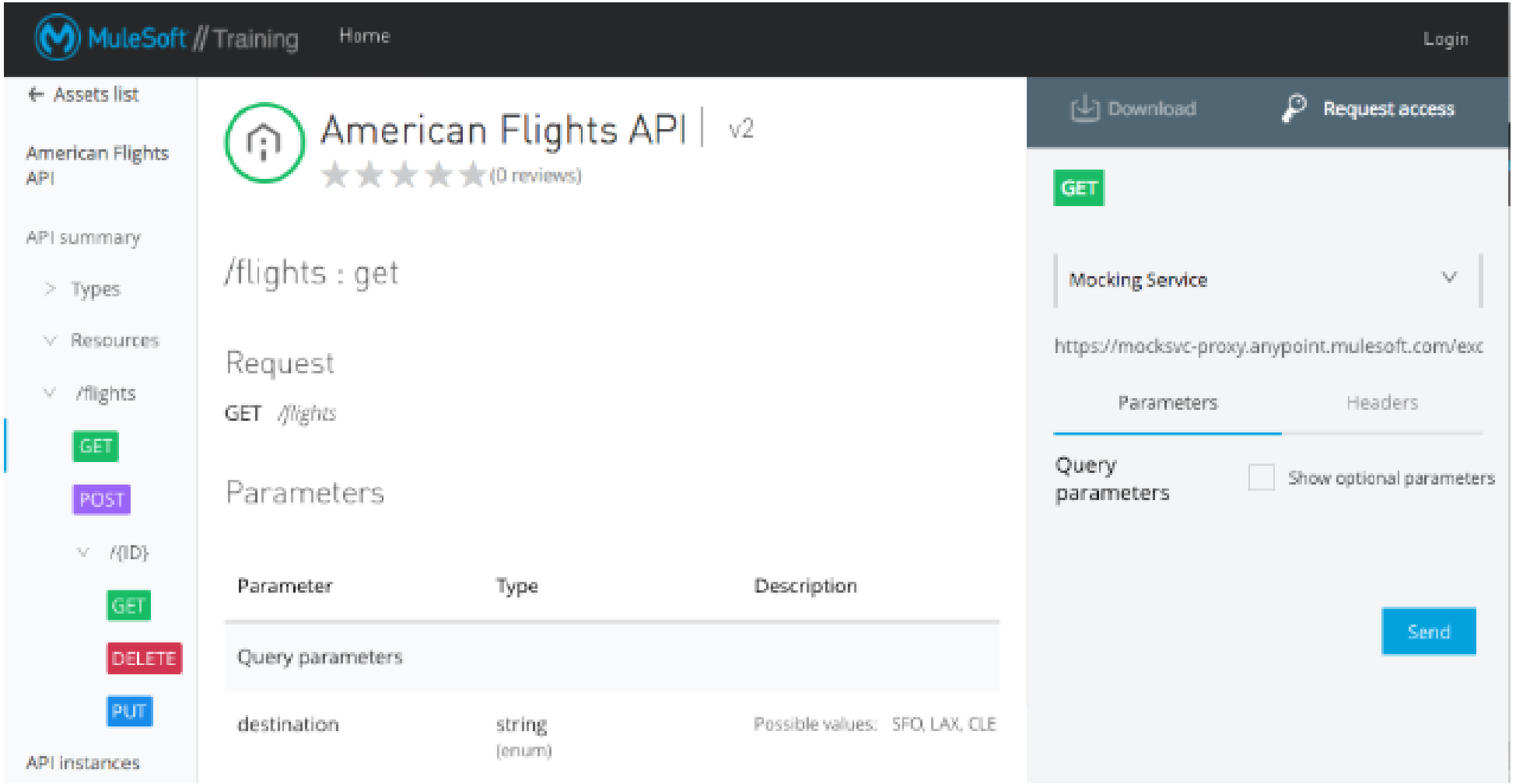


15. Click the American Flights API.
16. Browse the API Summary on the left-side and discover the resources that are available for the API.



17. Select the GET method.

18. Review the information about the GET method; you should see there is an optional query parameter called destination.



19. Scroll down and review the Headers and Response sections.

Headers

Parameter	Type	Description
client_id (required)	string	
client_secret (required)	string	

Response

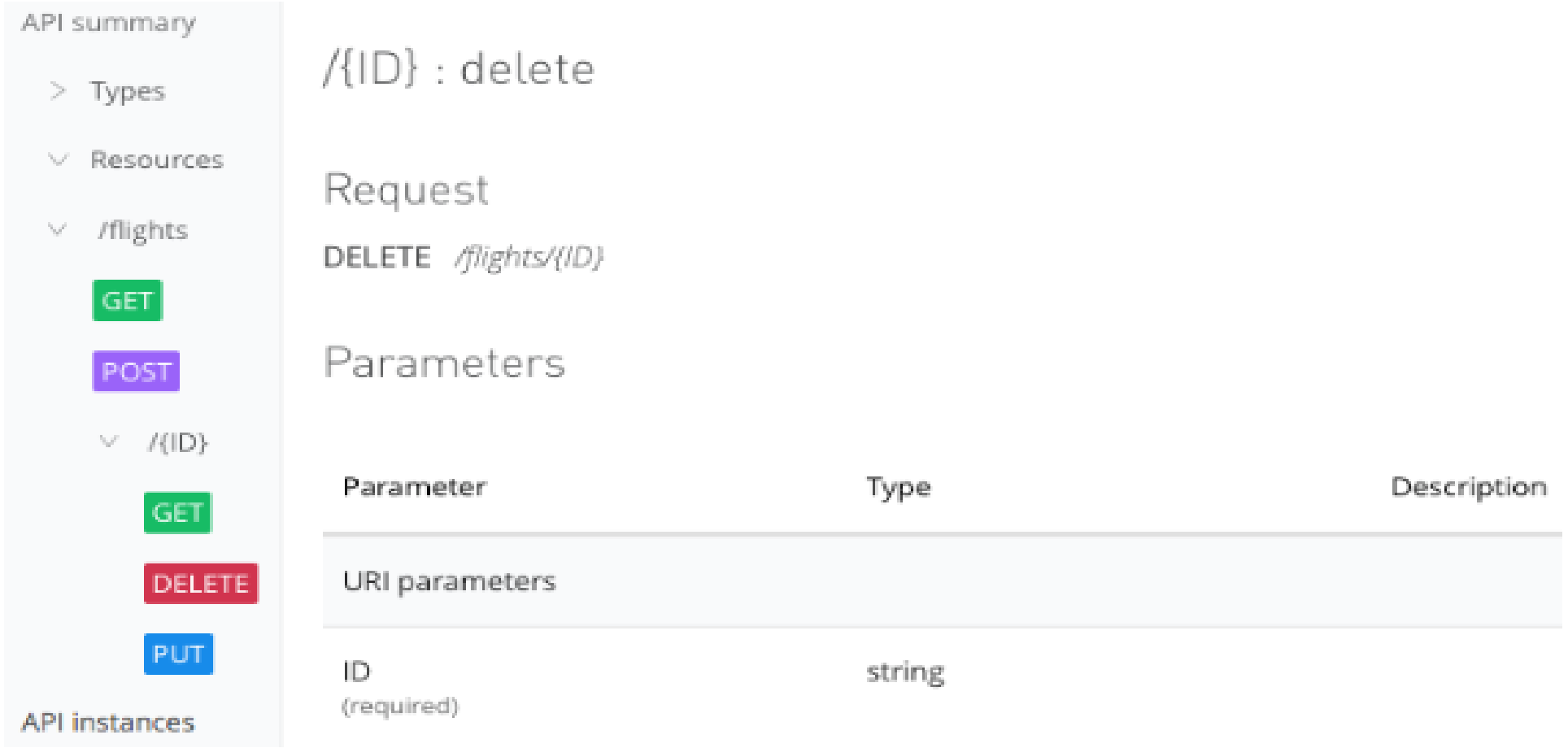
200

Typeapplication/json

Type | Examples

```
[
  {
    "ID": "integer",
    "code": "string",
    "price": "number",
    "departureDate": "string",
    "origin": "string",
```

20. In the left-side navigation, select the DELETE method.
21. Locate information about the required ID URI parameter.



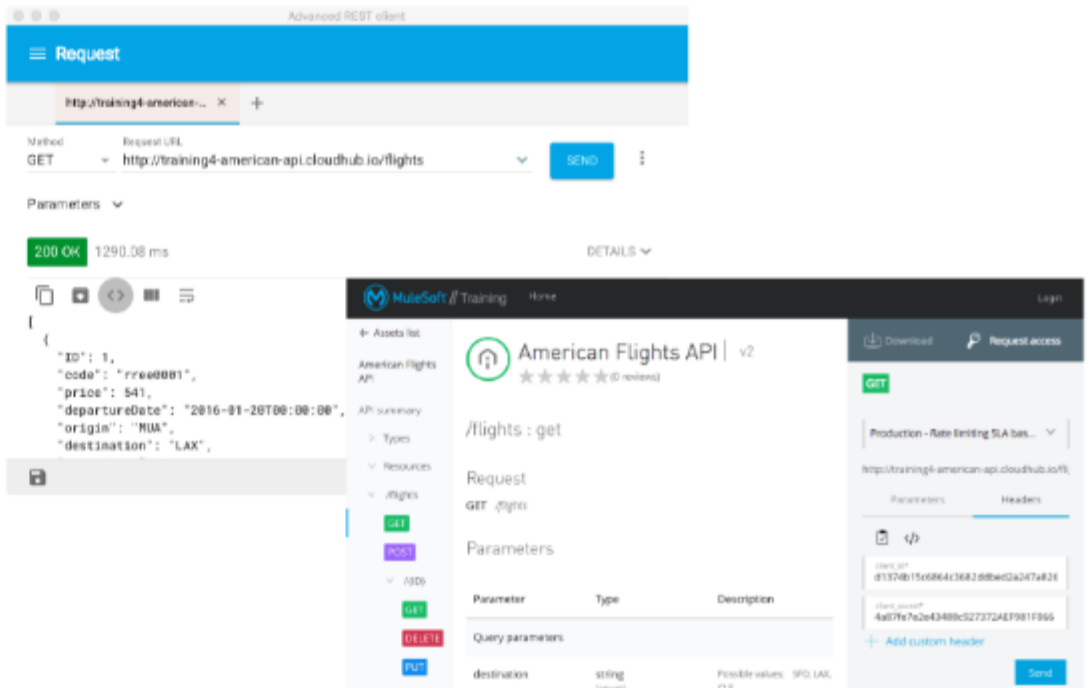
22. Review the information for the other resources.

Calling RESTful web services

Calling RESTful web services

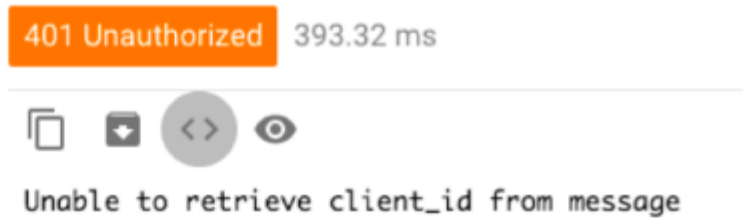
- To call web services, you need to write code or have a tool to make the HTTP requests
  - Need to be able to specify the HTTP method, request headers, and request body

- Example tools include
  - An API portal with an API console
  - Advanced Rest Client
  - Postman
  - A cURL command-line utility



Making calls to RESTful APIs

- **Unsecured APIs**
  - The API may be public and require no authentication



- **Secured APIs**
  - The API may be secured and require authentication
  - You may need to provide credentials and/or a token
  - Often a proxy is created to govern access to an API
  - We will call and then later create an API secured by credentials
  - You can also secure an API with other authentication protocols
    - OAuth, SAML, JWT, and more

Getting responses from web service calls

- RESTful web services return an HTTP status code with the response
- The status code provides client feedback for the outcome of the operation (succeeded, failed, updated)
  - A good API should return status codes that align with the HTTP spec

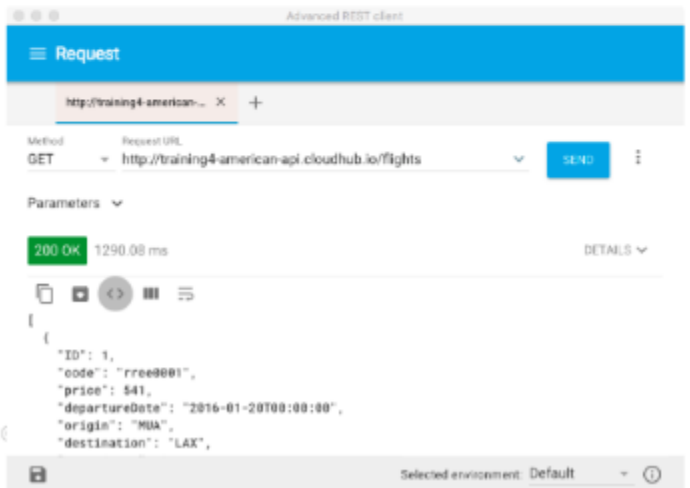
Common HTTP status codes

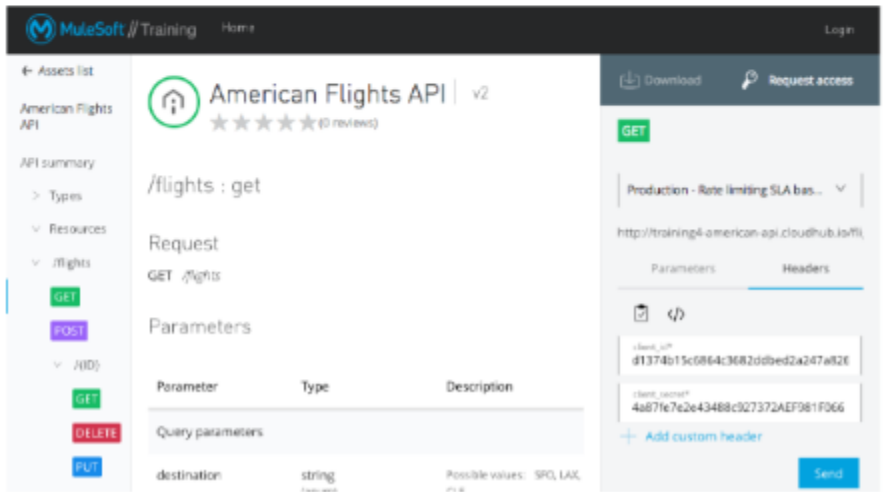
MuleSoft

Code	Definition	Returned by
200	OK – The request succeeded	GET, DELETE, PATCH, PUT
201	Created – A new resource or object in a collection	POST
304	Not modified – Nothing was modified by the request	PATCH, PUT
400	Bad request – The request could not be performed by the server due to bad syntax or other reason in request	All
401	Unauthorized – Authorization credentials are required or user does not have access to the resource/method they are requesting	All
404	Resource not found – The URI is not recognized by the server	All
500	Server error – Generic something went wrong on the server side	All

Walkthrough 1-2: Make calls to an API

- Use ARC to make calls to an unsecured API (an implementation)
- Make GET, DELETE, POST, and PUT calls
- Use ARC to make calls to a secured API (an API proxy)
- Use the API console in an API portal to make calls to a managed API using a mocking service
- Use the API console to make calls to an API proxy endpoint

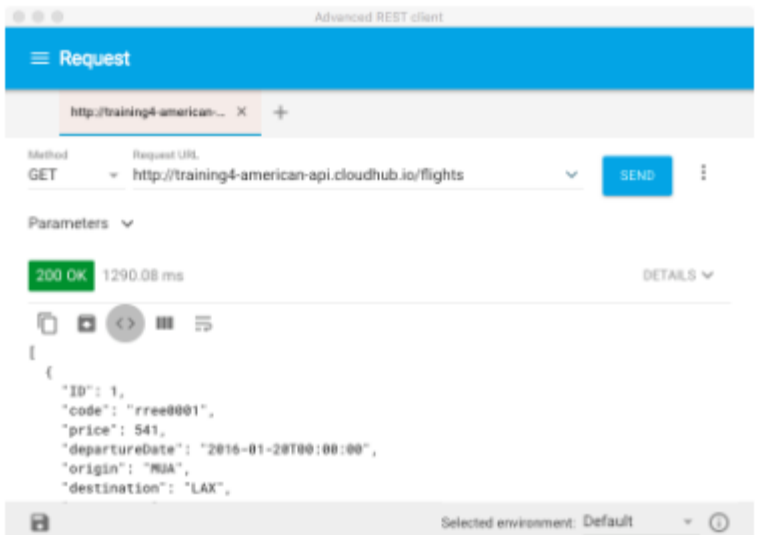


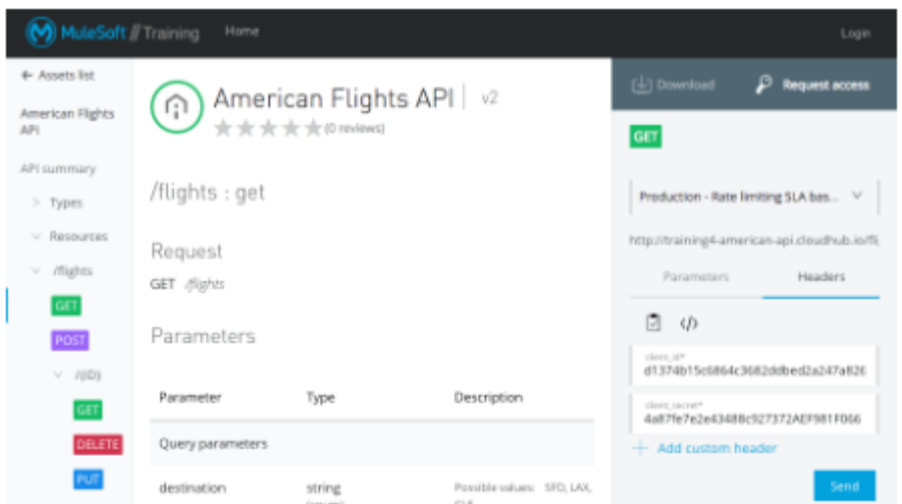


Walkthrough 1-2: Make calls to an API

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

- Use Advanced REST Client to make calls to an unsecured API (an implementation).
- Make GET, DELETE, POST, and PUT calls.
- Use Advanced REST Client to make calls to a secured API (an API proxy).
- Use the API console in an API portal to make calls to a managed API using a mocking service.
- Use the API console to make calls to an API proxy endpoint.

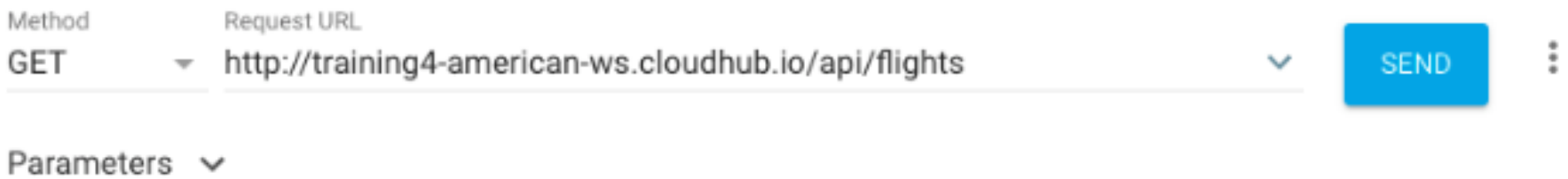




Use Advanced REST Client to make GET requests to retrieve data

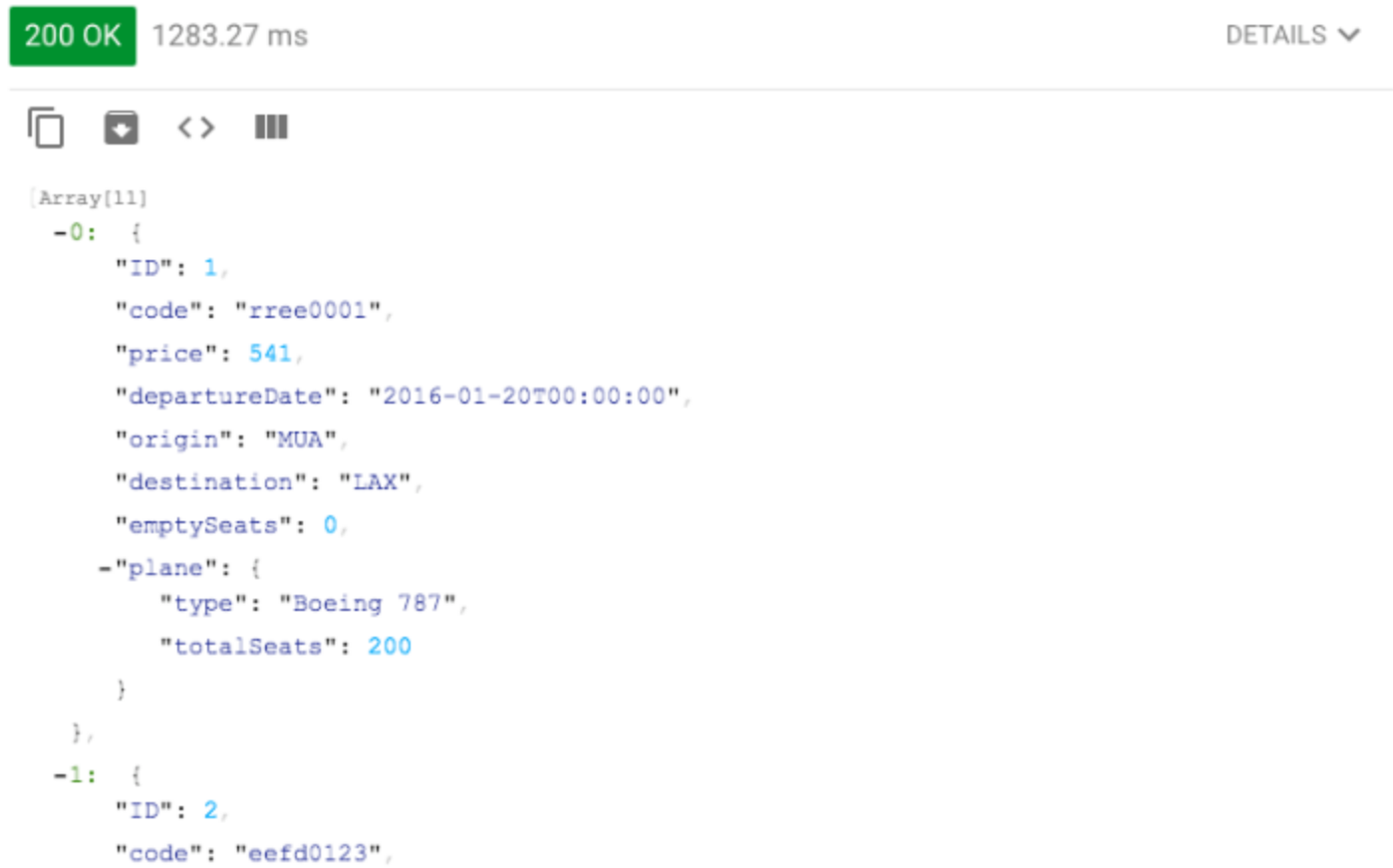
1. Return to or open Advanced REST Client.
2. Make sure the method is set to GET.
3. Return to the course snippets.txt file.
4. Copy the URL for the American Flights web service:
   http://training4-american-ws.cloudhub.io/api/flights.

   Note: This is the URL for the API implementation, not the managed API proxy. The -ws stands for web service.
5. Return to Advanced REST Client and paste the URL in the text box that says Request URL.



6. Click the Send button; you should get a response.
7. Locate the return HTTP status code of 200.

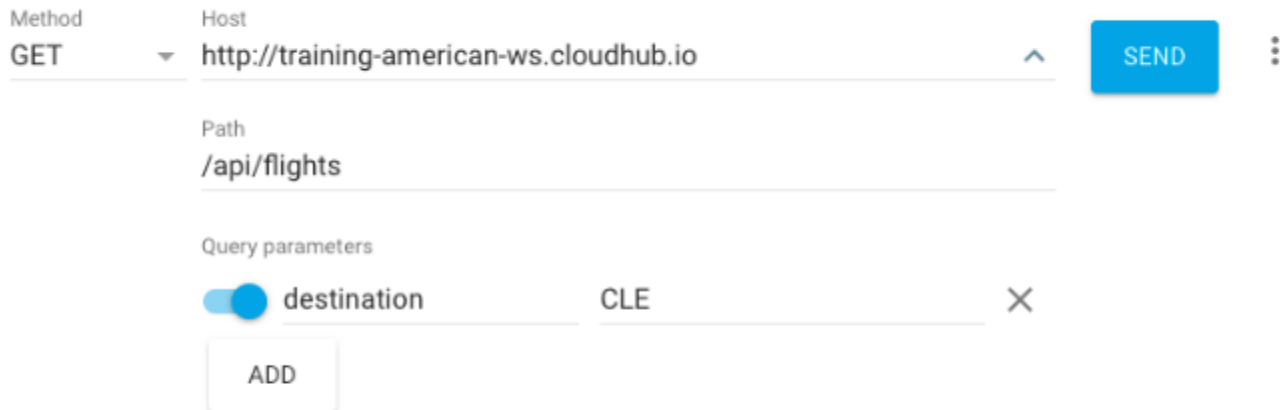
8. Review the response body containing flights to SFO, LAX, and CLE.



9. Click the Toggle raw response view button.

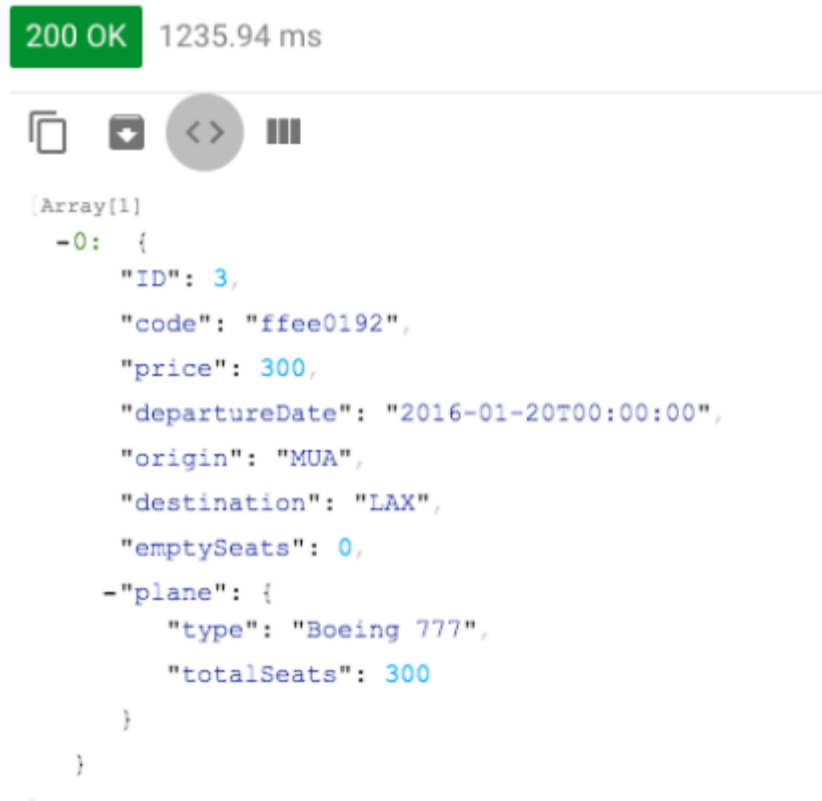
10. Click the arrow to the right of the URL.

10. Click the arrow to the right of the URL.
11. In the area that appears, set the Param name to destination and the value to CLE.



12. Click the Send button; you should get just flights to CLE returned.
13. Click the X next to the parameter to delete it.
14. Click the arrow to the right of the URL to collapse the parameters section.
15. Change the request URL to use a uri parameter to retrieve the flight with an ID of 3: http://training4-american-ws.cloudhub.io/api/flights/3.

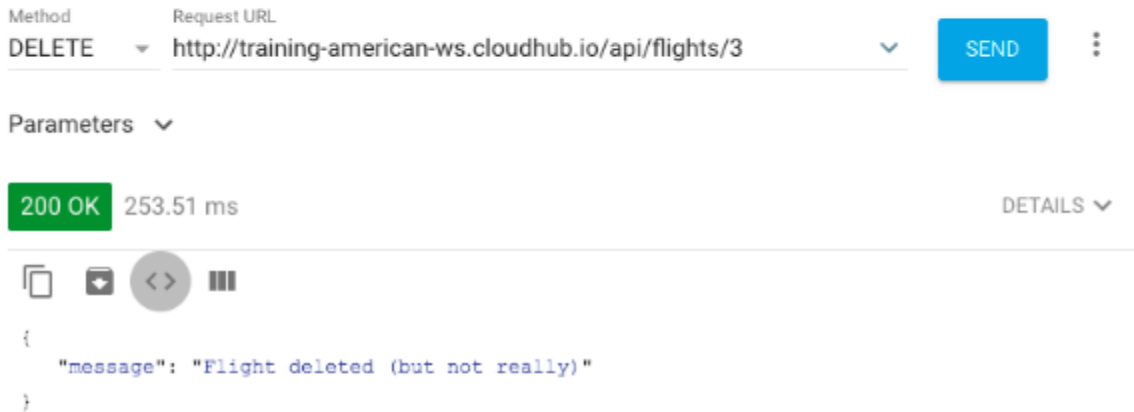
16. Click the Send button; you should see only the flight with that ID returned.



Make DELETE requests to delete data

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

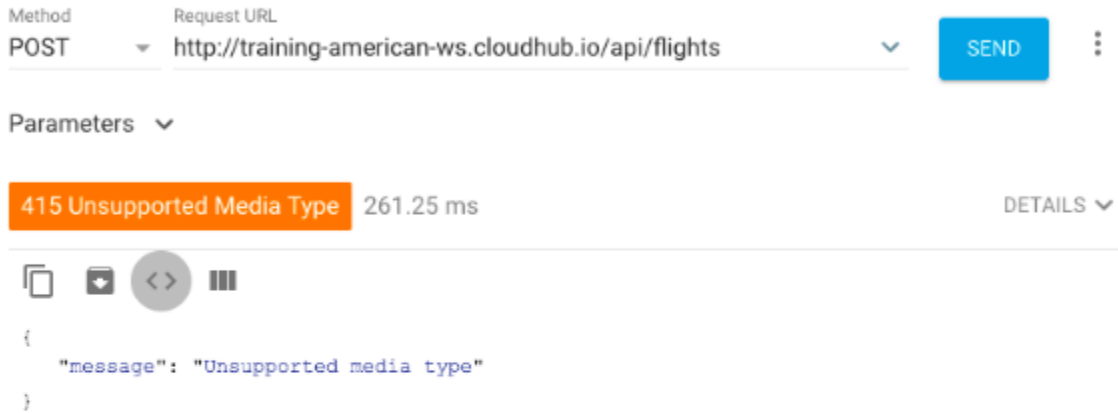
    Note: The database is not actually modified so that its data integrity can be retained for class.



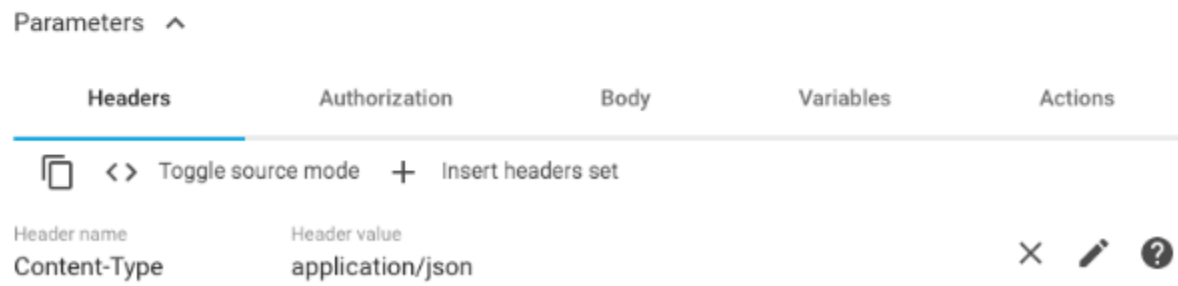
19. Remove the URI parameter from the request: http://training4-american-ws.cloudhub.io/api/flights.
20. Click the Send button; you should get a 405 response with a message of method not allowed.

Make a POST request to add data

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

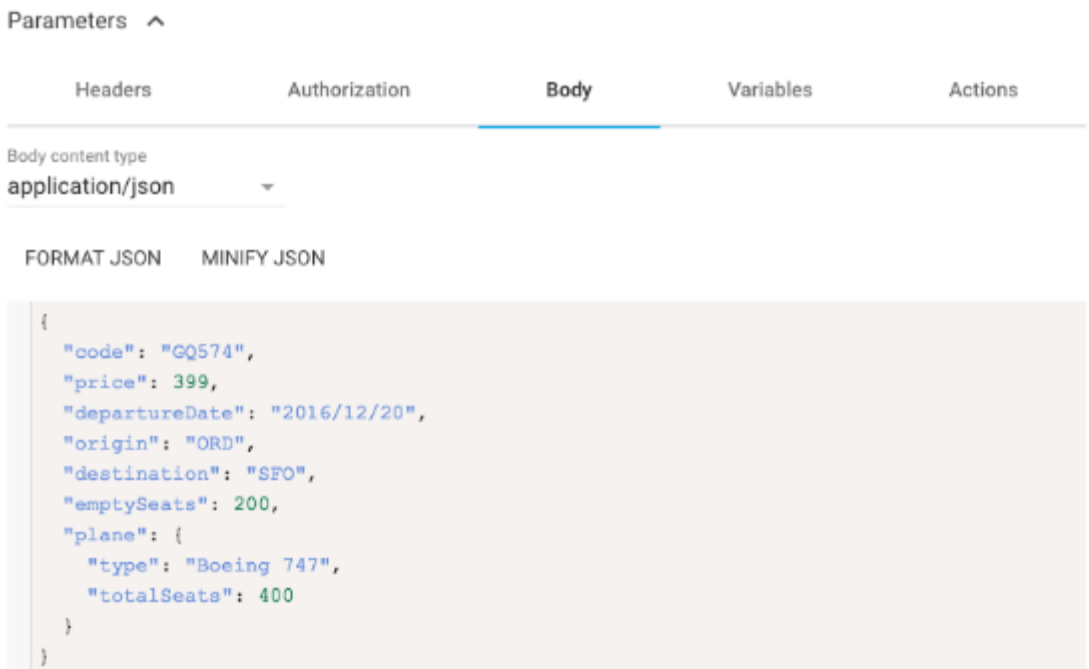


23. Click the arrow next to Parameters beneath the request URL.
24. Click in the Header name field, type C, and then select Content-Type.
25. In the Header value field, select application/json.



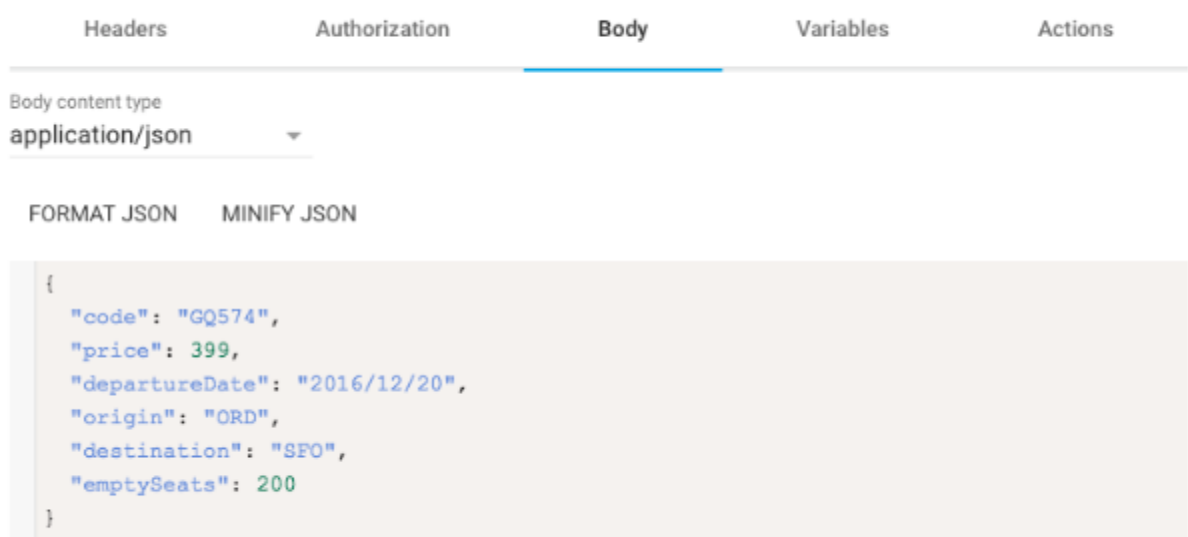
26. Select the Body tab.
27. Return to the course snippets.txt file and copy the value for American Flights API post body.

28. Return to Advanced REST Client and paste the code in the body text area.



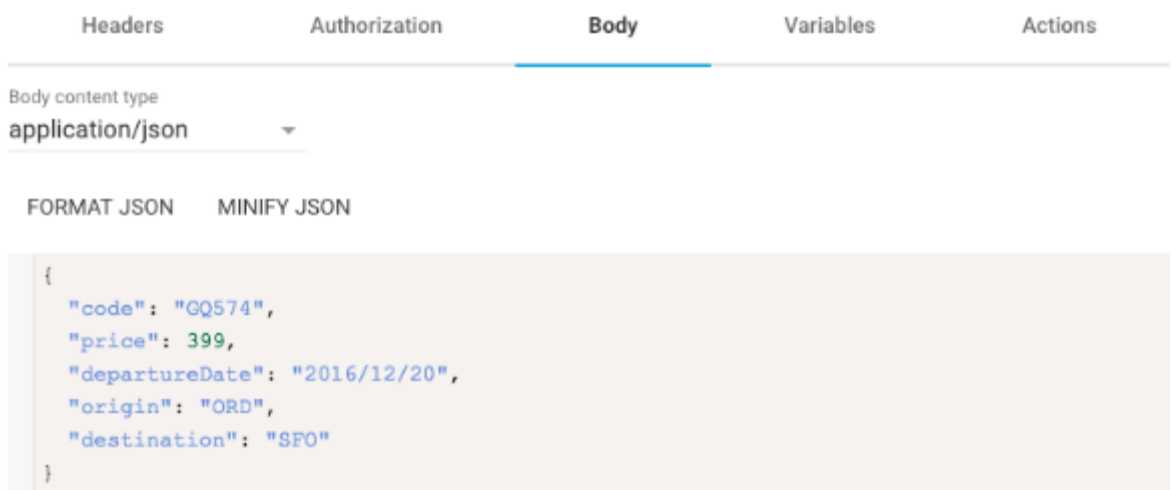
29. Click the Send button; you should see a 201 Created response with the message Flight added (but not really).

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



32. Send the request; the message should still post successfully.
33. In the request body, remove the emptySeats key/value pair.

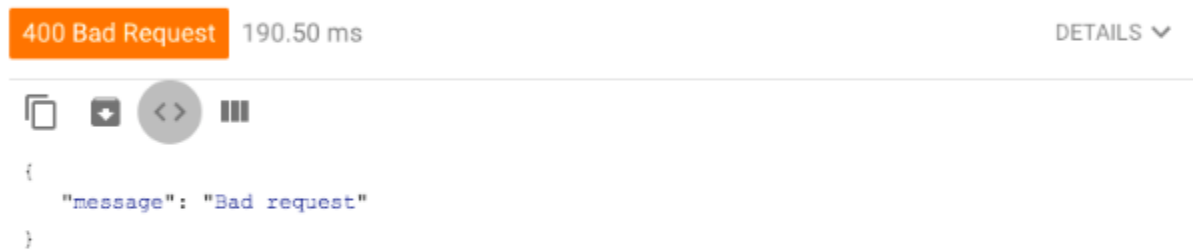
34. Delete the comma after the destination key/value pair.



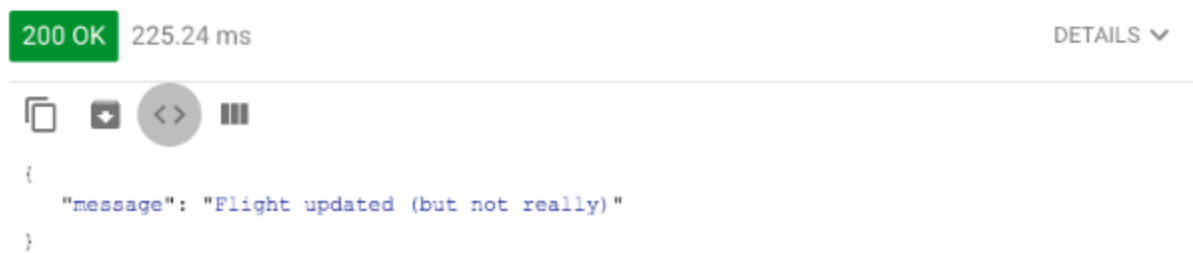
35. Send the request; you should see a 400 Bad Request response with the message Bad request.

Make a PUT request to update data

36. Change the method to PUT.
37. Add a flight ID of 3 to the URL.
38. Click the Send button; you should get a 400 Bad Request.



39. In the request body field, press Cmd+Z or Ctrl+Z so the emptySeats field is added back.
40. Send the request; you should get a 200 OK response with the message Flight updated (but not really).

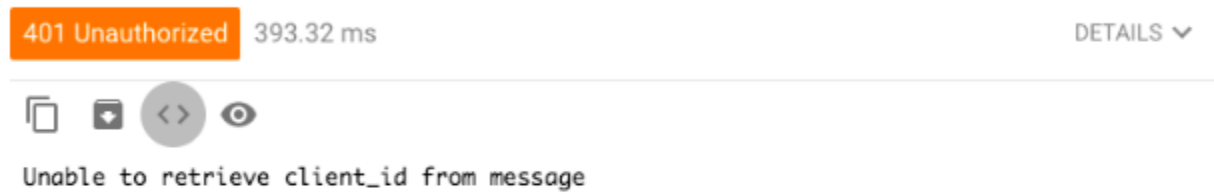


Make a request to a secured API

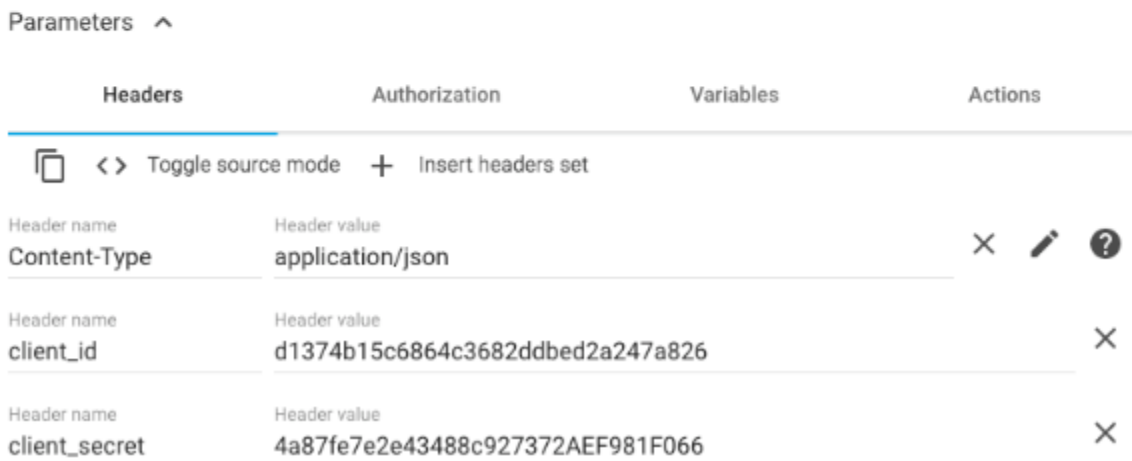
41. Change the method to GET.
42. Change the request URL to http://training4-american-api.cloudhub.io/flights/3.

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

43. Click the Send button; you should get a 401 Unauthorized response with a message about an invalid client_id or secret.



44. Return to the course snippets.txt file and copy the value for the American Flights API client_id.
45. Return to Advanced REST Client and add a header called client_id.
46. Set client_id to the value you copied from the snippets.txt file.
47. Return to the course snippets.txt file and copy the value for the American Flights API client_secret.
48. Return to Advanced REST Client and add a second header called client_secret.
49. Set client_secret to the value you copied from the snippets.txt file.



50. Click the Send button; you should get data for flight 3 again.

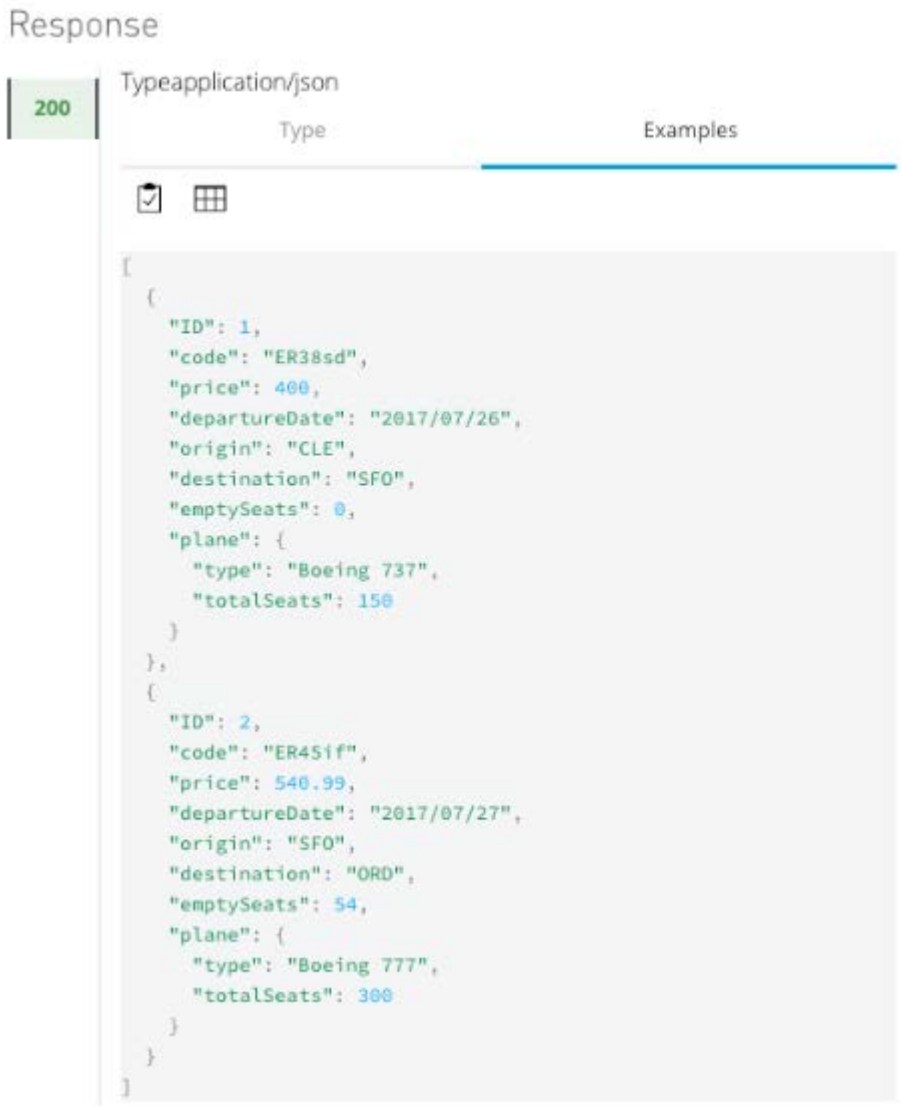
    Note: The API service level agreement (SLA) for the application with this client ID and secret has been set to allow three API calls per minute.

51. Click the Send button three more times; you should get a 429 Too Many Requests response with a Quota has been exceeded message.

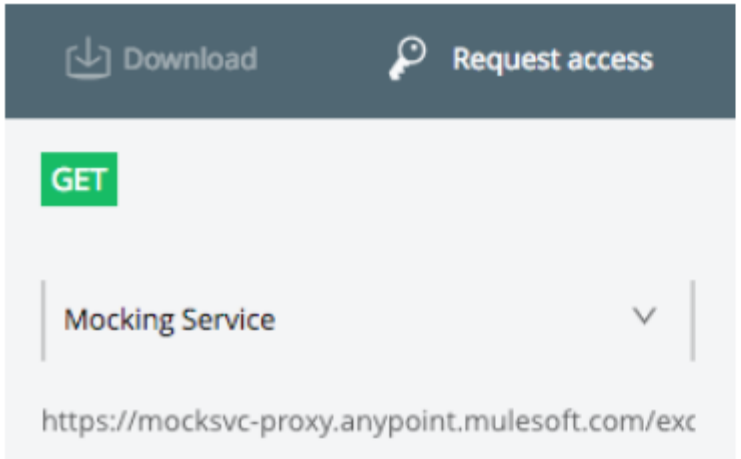
    Note: The API service level agreement (SLA) for the application with this client ID and secret has been set to allow three API calls per minute.

Use the API console in the API portal to make requests to the API using a mocking service

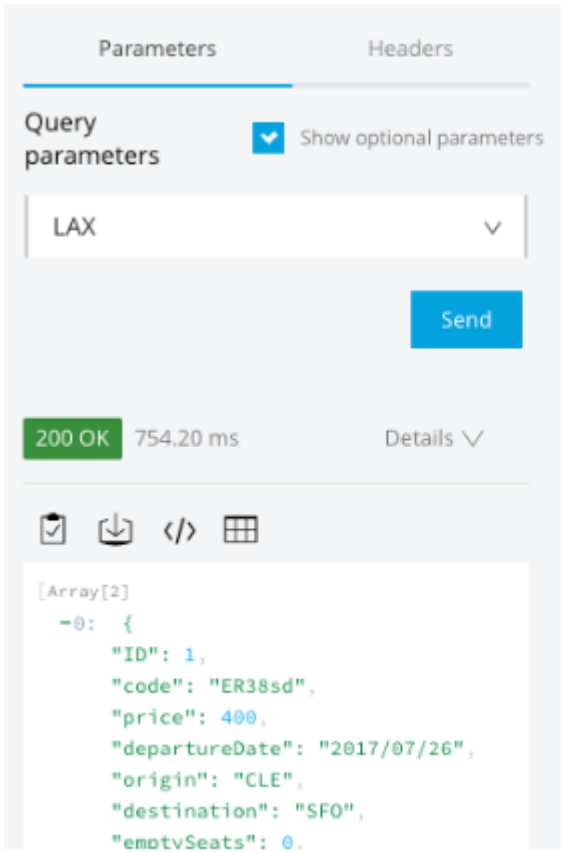
52. Return to the bowser window with the American Flights API portal at https://anypoint.mulesoft.com/exchange/portals/muletraining.
53. In the left-side navigation click the GET method for /flights.
54. Review the Headers and Response sections.
55. In the Response section, select Examples.



56. In the API console located on the right side of the page, make sure the Mocking Service endpoint is selected.
57. Look at the endpoint URL that is displayed.

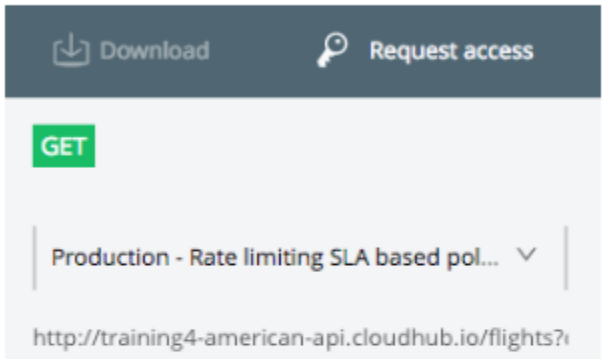


58. Click the Show optional parameters checkbox.
59. Select LAX in the destination drop-down menu.
60. Click Send.
61. Look at the response; you should get the example flights that are to SFO.

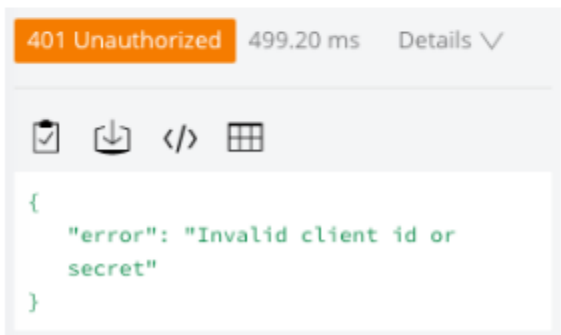


Make requests to the API using an API proxy endpoint

62. At the top of the API console, change the endpoint to Production – Rate limiting SLA based policy.
63. Look at the endpoint URL that is displayed.

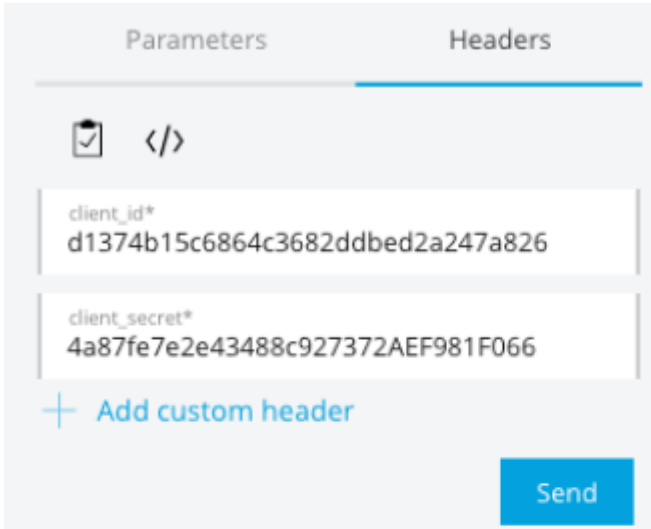


64. Click Send; you should get a 401 Unauthorized response.

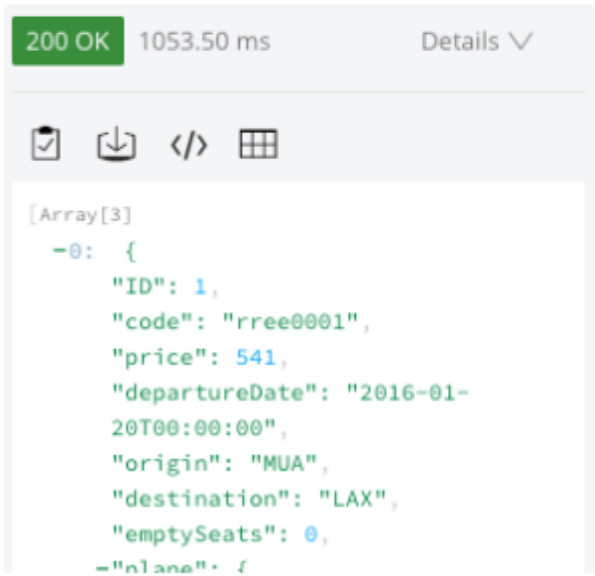


65. Select the Headers tab.

66. Copy and paste the client_id and cient_secret values from the course snippets.txt file.

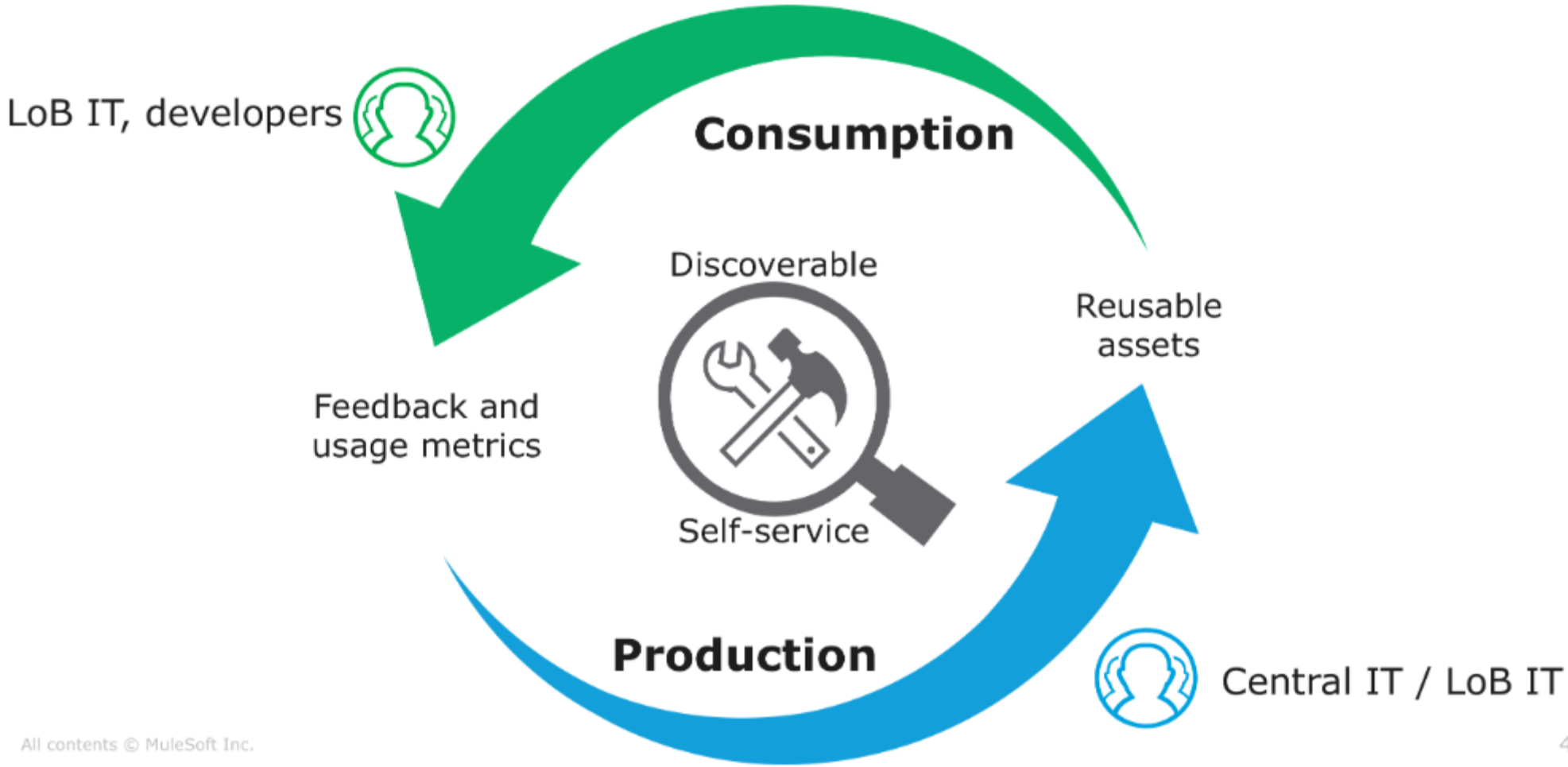


67. Click Send; you should get a 200 OK response with only flights to LAX.



Successfully creating application networks using API-led connectivity

Producing discoverable and consumable assets is key



Designing for API success

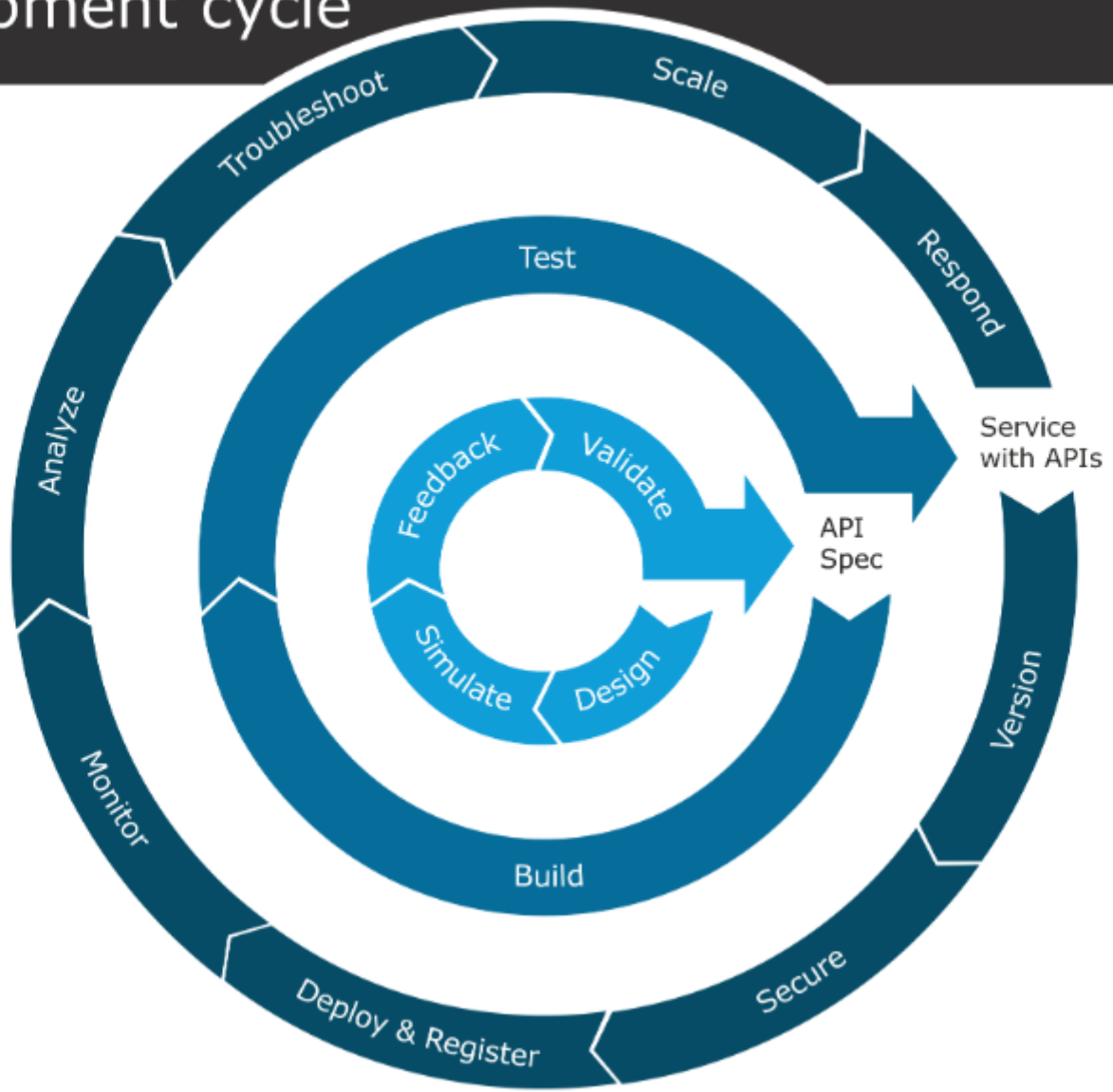
MuleSoft

- Create APIs that developers **can find** and **want to use** and share with others
  - Design the API for the business use cases it will fulfill, not to model the backend services or applications they expose
  - Focus on performance of client applications and user experience

- Take an **API design-first approach**!

- **Get API design right** before investing in building it
  - Define it iteratively getting feedback from developers on its usability and functionality along the way
  - Building the implementation of an API is time consuming and expensive to undo

API development cycle



Summary

Summary

MuleSoft

- Companies today need to **rapidly adopt and develop** new technologies in order to stay relevant to customers & keep competitive
- IT needs to be able to rapidly integrate resources and make them **available for consumption**
  - An **API-led connectivity** approach can help achieve this
- To drive API-led connectivity, create a **C4E** (Center for Enablement)
  - A cross-functional team to ensure assets across the organization are productized, published, and widely consumed
- **An application network** is a network of applications, data, and devices connected with APIs to make them pluggable and to create reusable services

Summary

MuleSoft

- A **web service** is a method of communication that allows two software systems to exchange data over the internet
- An **API** is an application programming interface that provides info for how to communicate with a software component
- The **term API** is often used to refer to any part of RESTful web service
  - The web service API (definition or specification file)
  - The web service interface implementing the API
  - The web service implementation itself
  - A proxy for the web service to control access to it
- **RESTful** web services use standard HTTP protocol and are easy to use
  - The HTTP request method indicates which operation should be performed on the object identified by the URL

Module 1: Introducing Application Networks and API-Led Connectivity

At the end of this module, you should be able

- 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
- Make calls to unsecured and secure APIs