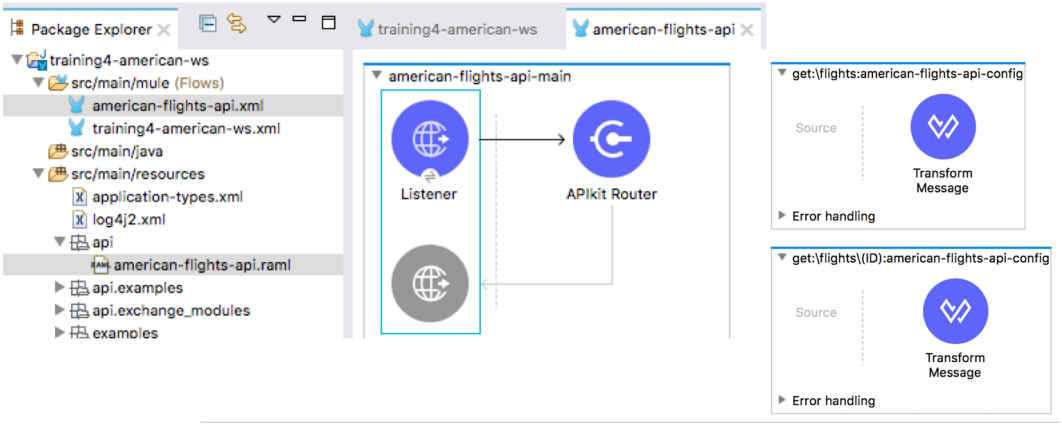
# Walkthrough 4-5: Use Anypoint Studio to create a RESTful API interface from a RAML file

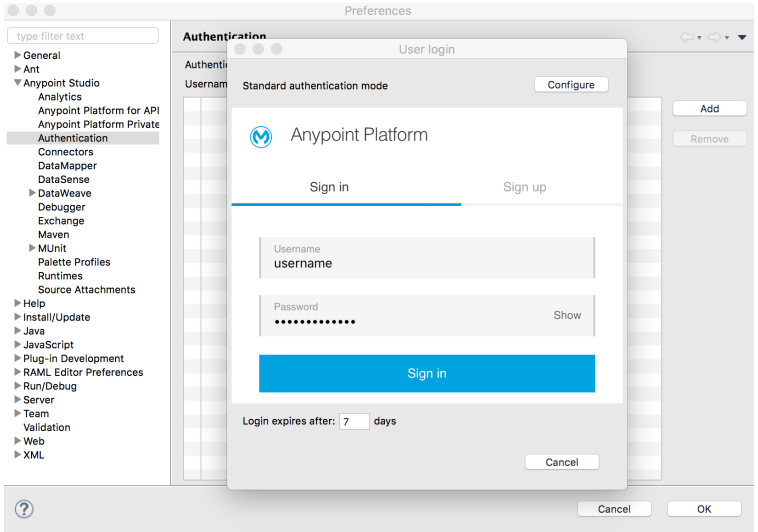
In this walkthrough, you generate a RESTful interface from the RAML file. You will:

- Add Anypoint Platform credentials to Anypoint Studio.
- Import an API from Design Center into an Anypoint Studio project.
- Use APIkit to generate a RESTful web service interface from an API.
- Test a web service using APIkit console and Advanced REST Client.




## Add Anypoint Platform credentials to Anypoint Studio

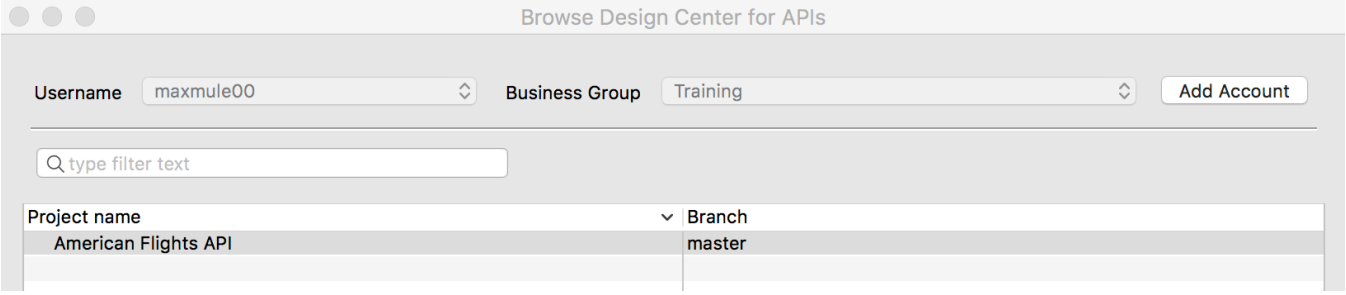
1. In Anypoint Studio, right-click training4-american-ws and select Anypoint Platform > Configure Credentials.
2. In the Authentication page of the Preferences dialog box, click the Add button.
3. In the Anypoint Platform Sign In dialog box, enter your username & password and click Sign In.



4. On the Authentication page, make sure your username is listed and selected.
5. Click OK.

## Add an API from Design Center to the Anypoint Studio project

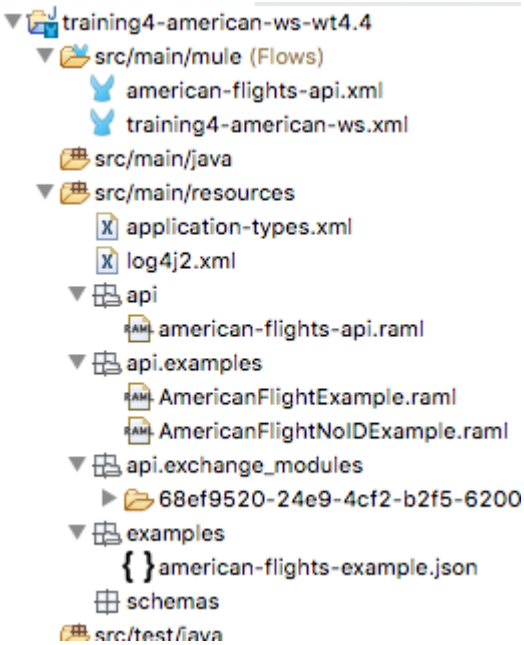
6. In the Package Explorer, locate the src/main/resources/api folder; it should not contain any files.
7. Right-click the folder (or anywhere in the project in the Package Explorer) and select Anypoint Platform > Import from Design Center.
8. In the Browse Design Center for APIs dialog box, select the American Flights API and click OK.




9. In the Override files dialog box, click Yes.

## Locate the API files added to the project

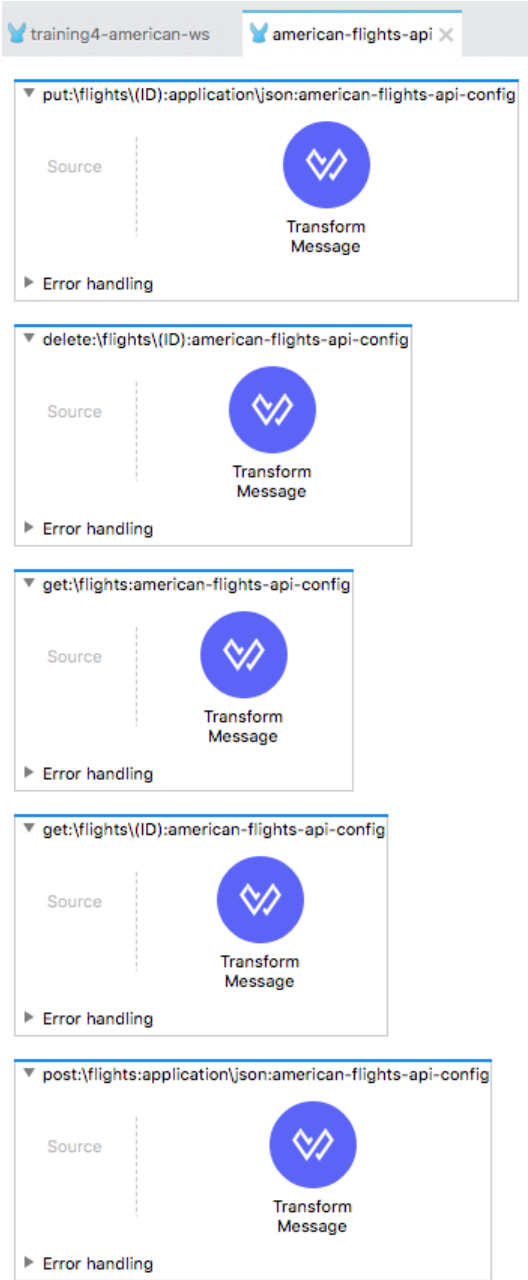
10. In the Package Explorer, locate and expand the src/main/resources folder; it should now contain api folders.
11. Expand the api folder; you should see the RAML file imported from Design Center.



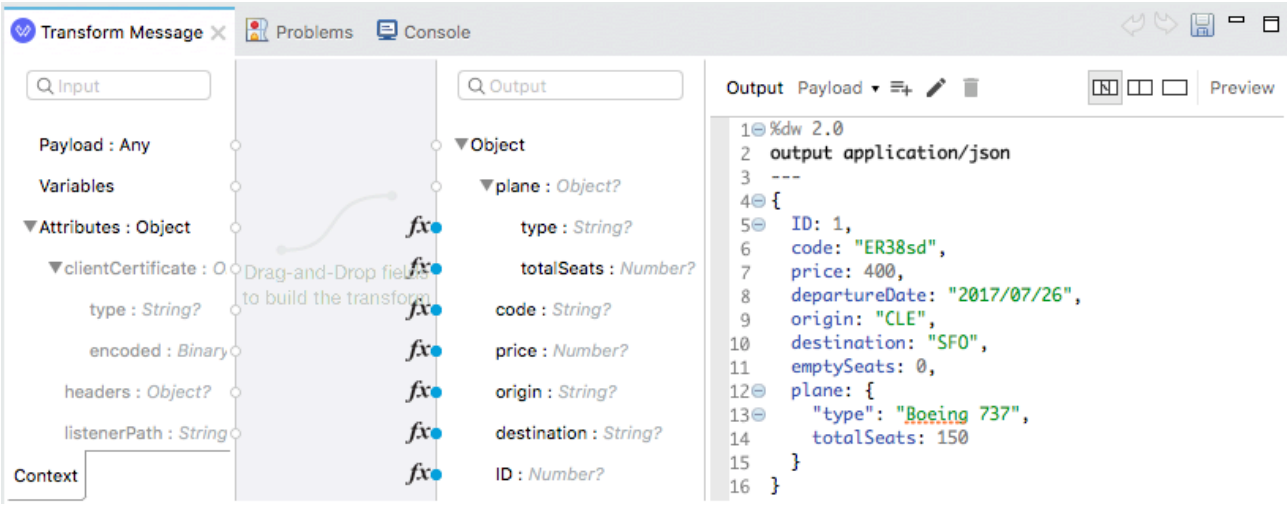
## Examine the XML file created

12. Examine the generated american-flghts-api.xml file and locate the following five flows:

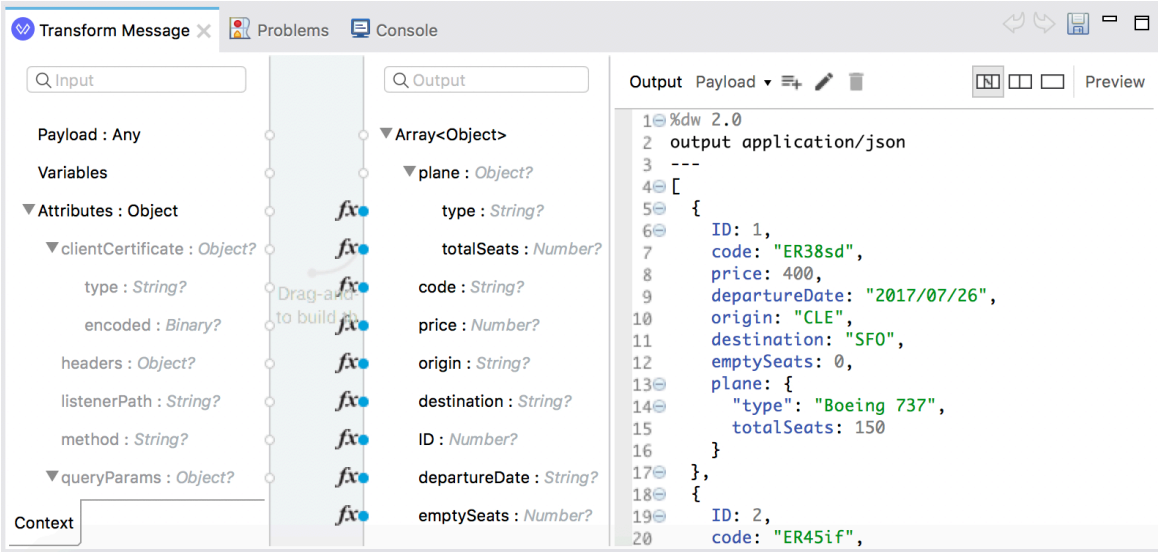
- get:/flights
- get:/flights/{ID}
- post:/flights
- delete:/flights/{ID}
- put:/flights/{ID}



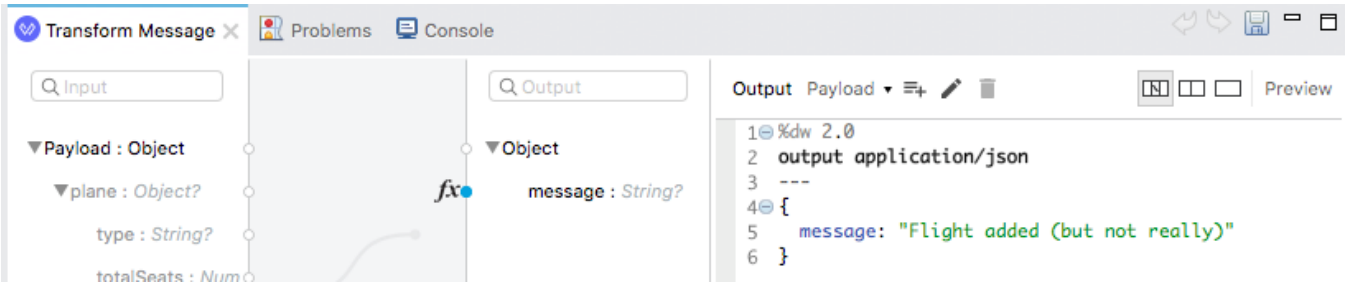
13. In the get:/flights/{ID} flow, double-click the Transform Message component and look at the value in the properties view.




14. In the get:/flights flow, double-click the Transform Message component and look at the output JSON in the properties view.

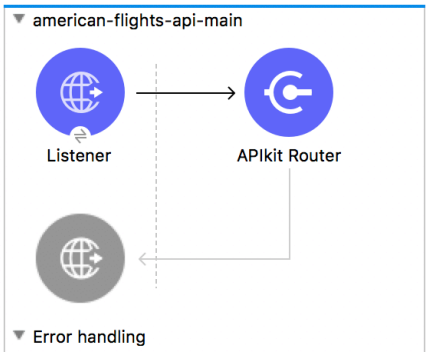



15. In the post:/flights flow, double-click the Set Payload transformer and look at the value in the Set Payload properties view.



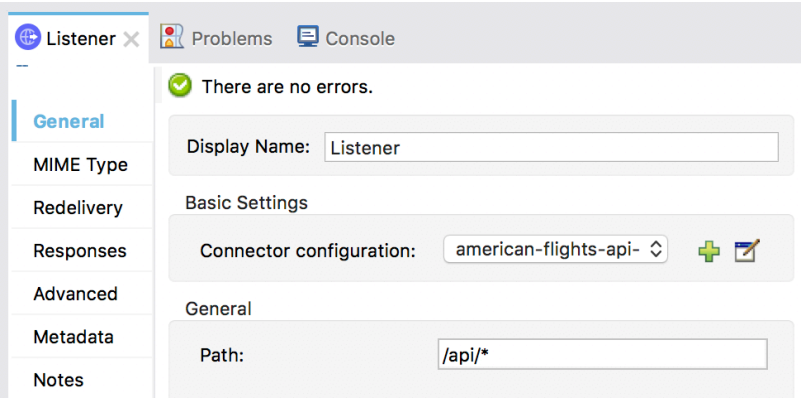
## Examine the main flow and the new HTTP Listener endpoint

16. Locate the american-flights-api-main flow.
17. Double-click its HTTP Listener.




18. In the Listener properties view, notice that the path is set to /api/*.

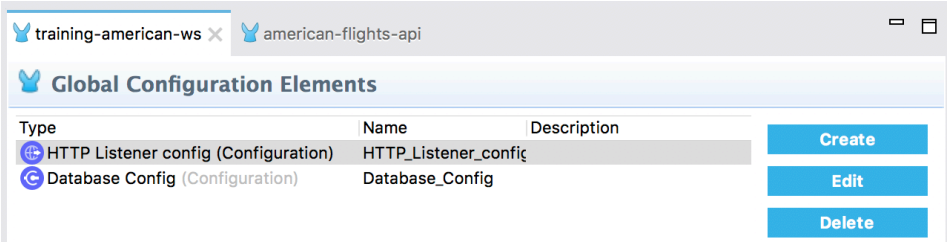
    *Note: The * is a wildcard allowing any characters to be entered after /api/.*



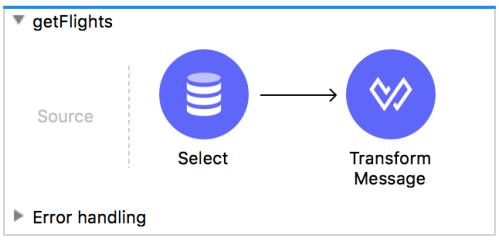

19. Click the Edit button for the connector configuration; you should see that the same port 8081 is used as the HTTP Listener you created previously.
20. Click OK.

## Remove the other HTTP configuration and listeners

21. Return to training4-american-ws.xml.
22. In the Global Elements view, select the HTTP Listener config and click Delete.



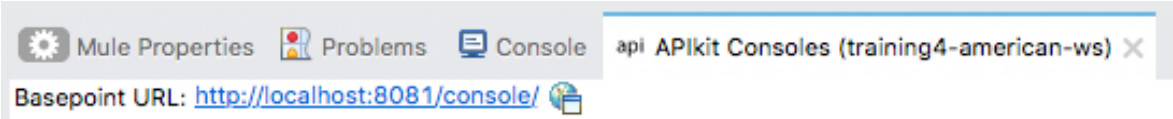
23. Return to the Message Flow view.
24. Right-click the HTTP Listener in getFlights and select Delete.



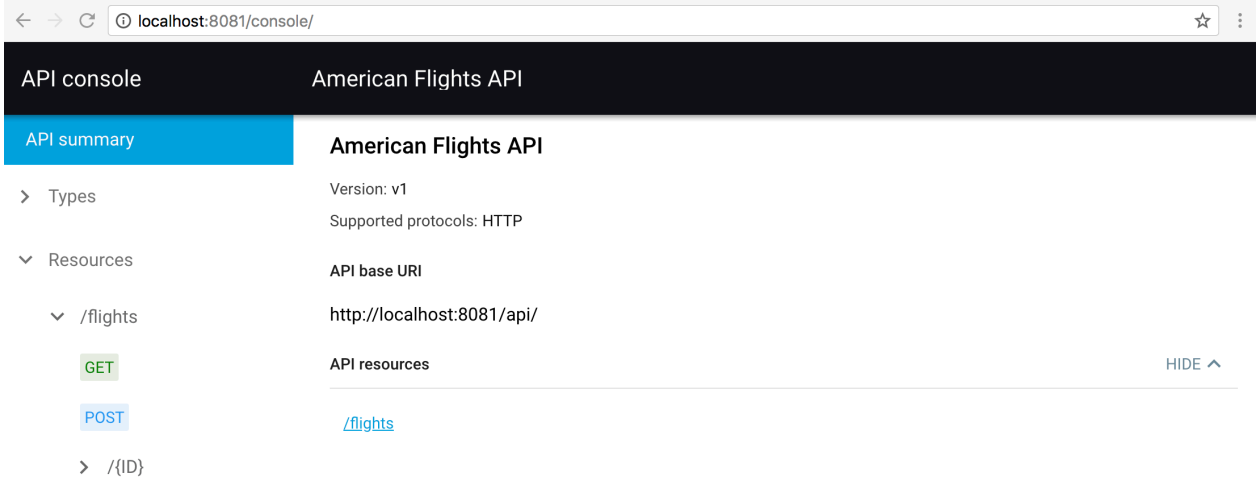

25. Delete the other two HTTP Listeners.

## Test the web service using APIkit console

26. Save the files.
27. Look at the console; you should see the application was not redeplpoyed.
28. Stop the project.
29. Run the project and wait until Mule and the application restart.
30. Locate the new APIkit Consoles view that is created and opened in Anypoint Studio.




31. Click the console link; a browser window should be open with an API console.



32. Select the GET method and click the TRY IT button.

API console | American Flights API

API summary
Types
Resources
/flights
GET
POST
/{ID}

/flights : **get**

TRY IT

**Request**

GET *http://localhost:8081/api/flights*

Parameters HIDE

| Parameter | Type | Description |
|---|---|---|
| Query parameters | | |
| **destination** | **string** (enum) | Possible values: **SFO, LAX, CLE** |

33. Click Send; you should get a 200 response with the example flight data – not all the flights.

API console | American Flights API

API summary
Types
Resources
/flights
GET
POST
/{ID}

Request URL
http://localhost:8081/api/flights

**Parameters** | **Headers**

Query parameters ☐ Show optional parameters

SEND

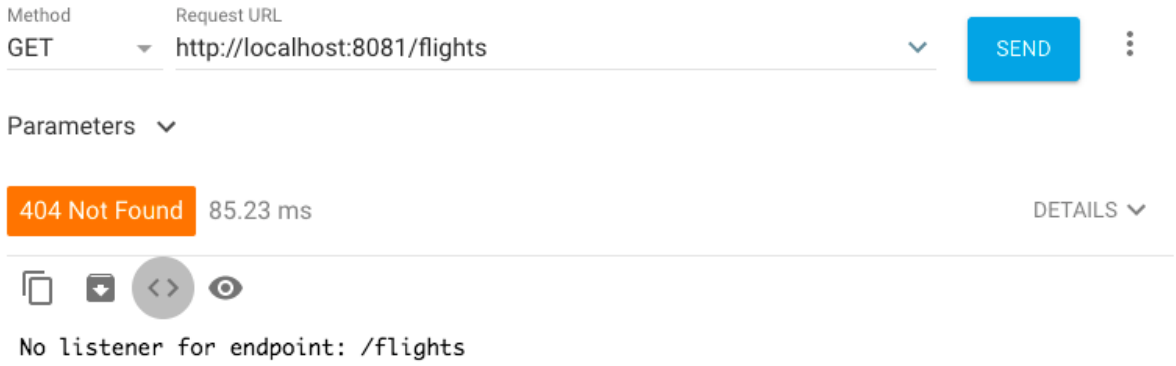
200 OK 417.50 ms DETAILS


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

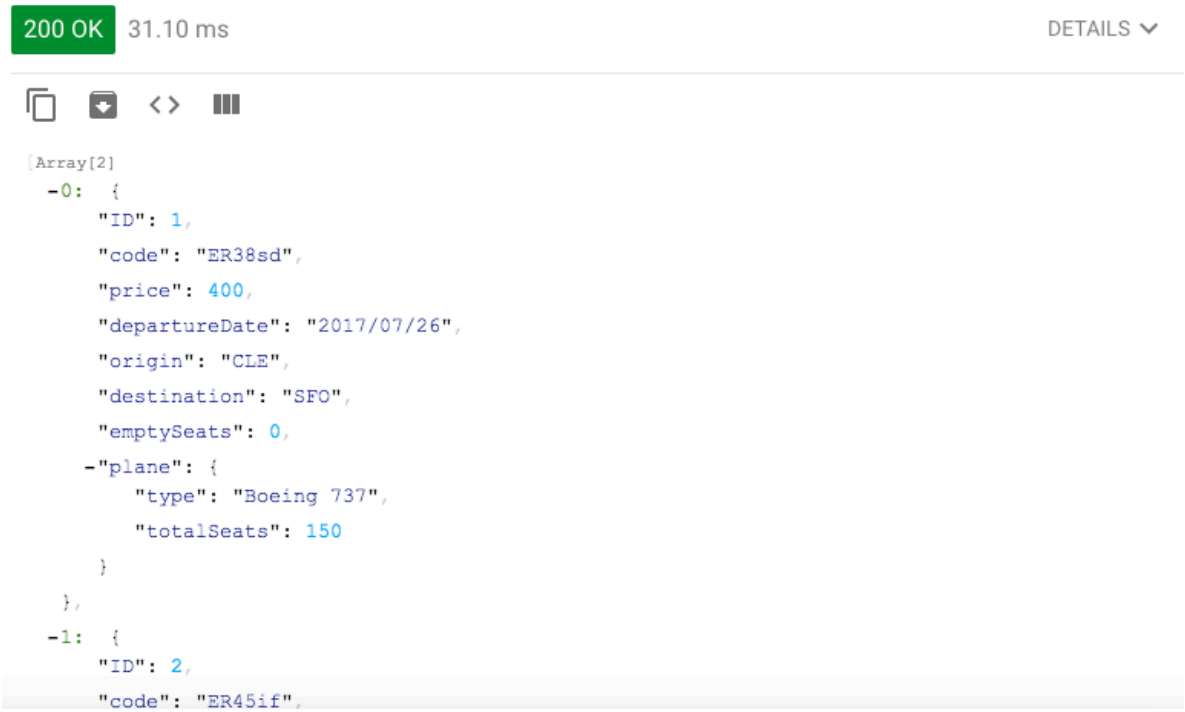

34. Close the browser window.

## Test the web service using Advanced REST Client

35. Return to Advanced REST Client.
36. Change the method to GET and click Send to make a request to http://localhost:8081/flights; you should get a 404 Not Found response.




37. Change the URL to http://localhost:8081/api/flights and send the request; you should get a 200 response with the example flight data.



38. Make a request to http://localhost:8081/api/flights/3; you should see the example data returned for a flight with an ID of 1.

200 OK 45.04 ms DETAILS


```
{
    "ID": 1,
    "code": "ER38sd",
    "price": 400,
    "departureDate": "2017/07/26",
```