Assignment Five

Purpose

The purpose of this assignment is to leverage Apigee's analytics policies to gather information about the requests being sent in by users.

Using the information already entered to the BaaS for the previous assignment, you will add another collection of reviews that are tied to the movies. This way users can query the BaaS and get the previous information (title, year released and actors) as well as the reviews. These two entities should remain separate! Do not append the reviews to the existing movie information.

Requirements

- Create a collection in Apigee's BaaS to hold reviews about existing movies.
  - A review contains the name of the reviewer, a small quote about what they thought about the movie, and their rating out of five stars.
  - The review collection should have at least one review for each movie.
  - The review can be a simple, fictitious review that you create.
- This proxy should build upon the previous proxy in assignment four.
  - If the user sends a response with the query parameter `reviews=true`, then the response should include the movie information as well as all the reviews for the movie. If they do not pass this in, the response should not show the reviews.
  - The review information should be appended to the response to the user.
- Add custom analytics to return information about which movies users are querying.
  - If you are using a127 or Node.js app, you will have to add the analytic policies *after* the proxy has been deployed.
  - Create a custom analytics policy that describes the number of times each movie has been queried. To do this, you will have to send a number of requests for each movie.

Acceptance Criteria

In previous assignments, not including a small test suite has had a minimal negative impact on grading. For this assignment, it is required. If a test suite is not included in your email or uploaded to GitHub, the assignment will not be graded.

- Create a SoapUI/Postman/collection of curl commands to test your proxy. You should include the following requests.
  - Valid request without the review query parameter.
  - Invalid request (for a movie not in the BaaS) without the review query parameter.
  - Valid request with the review query parameter.

  - A request to the BaaS which returns a movie (this request does not go through the proxy).
  - A request to the BaaS which returns a review (this request does not go through the proxy).

- Include the CSV results from your custom report generated by Apigee.
  - When viewing the custom report, you can export as a CSV.

Resources

You can connect to Apigee's BaaS through the dashboard (page you see when first signing into Edge) by selecting BaaS or `http://appservices.apigee.com`.

- `http://apigee.com/docs/app-services/content/connecting-users-other-data` Describes how to connect two entities from different BaaS collections together.
- Depending on how you implement retrieving reviews, you may find some of the following links helpful.
  - `https://github.com/caolan/async` If using a Node.js backend, you will need information from the initial response for the movie to make a subsequent callout to retrieve the reviews. Async is one tool for handling asynchronous calls.
  - `https://www.npmjs.com/package/usergrid` Apigee has also provided an SDK with an API to retrieve connected entities.
- `http://apigee.com/docs/analytics-services/content/analyze-api-message-content-using-custom` Information about adding Apigee's custom analytics to the proxy.