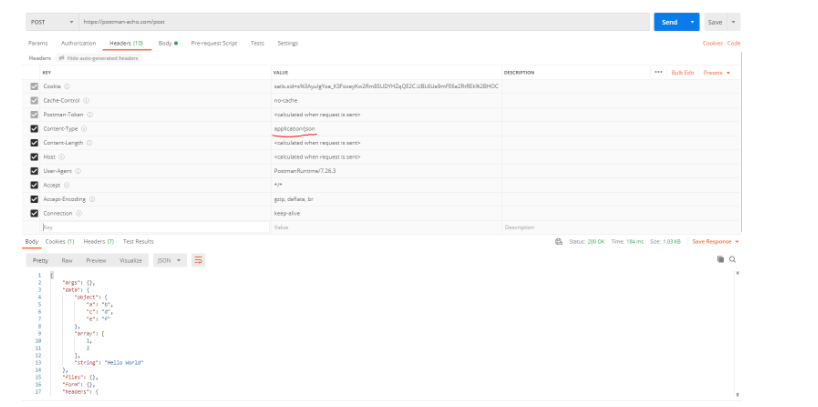
Week 2 - Lab Activity - Code - A Postman GET

In this lab you will develop a POST function that passes the JSON object (created this week in the DQ) to a server.

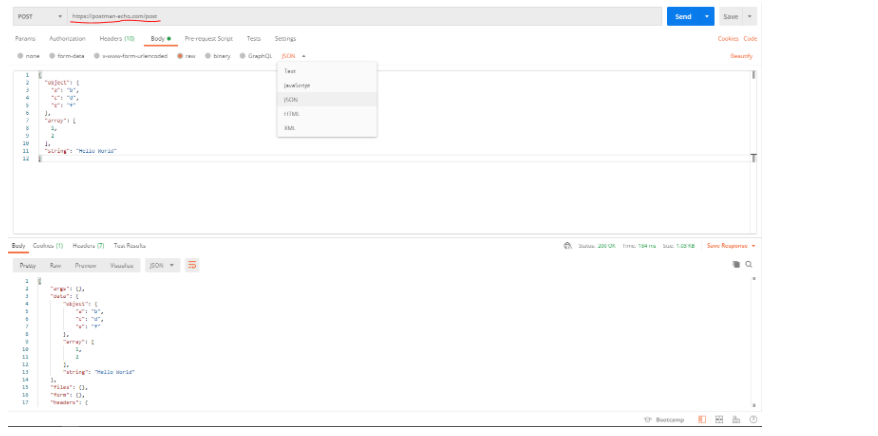
We will be using [Postman (Links to an external site.)](https://www.postman.com/) so it is highly recommended that you logon to the [Postman Learning center (Links to an external site.)](https://learning.postman.com/) and read the Getting Started Guide. If you have any questions please post them to [Week 2 - Questions](https://online.macomb.edu/courses/43853/discussion_topics/395651)

The following are some screen captures that I created to demonstrate creating a sample post to [https://docs.postman-echo.com/post (Links to an external site.)](https://docs.postman-echo.com/post). (This is a test URL which you can use to test your POST of the JSON object)

Screen capture of the 'content header'. Notice the "**Content-Type**" header is set to "**application/json**".



Screen capture of the JSON body. Notice the post URL is the Postman Echo test server. Also notice that the return data is formatted as JSON.



Submit 2 screen captures like the sample above but with your JSON object. Please make sure the POST is to the [https://docs.postman-echo.com/post (Links to an external site.)](https://docs.postman-echo.com/post) URL

**Scoring:**

* A screen shot of the successful POST (must have a return status of 200 OK) - 10pts.
* The JSON object in this week's DQ is used in the POST and is visible in the screen capture - 5pts.

[Previous](https://online.macomb.edu/courses/43853/modules/items/1995860)[Next](https://online.macomb.edu/courses/43853/modules/items/2022179)

Week 2 - Lab Activity

In this lab activity you are going to post a JSON object to a DynamoDb NOSQL database, read the item back, and then delete it.

Please watch the videos in this module before proceeding.

Note: You will also have to install the [Requests Python library (Links to an external site.)](https://requests.readthedocs.io/en/master/)

**Assignment:**

Create a Python program that does the following:

* Defines 3 functions:
  + **PostItem (url, item)** - This function has the parameters **url** as a string and **item** as a Python dictionary. The **item** that is passed to the function is a JSON object that represents something in your individual project. The function uses the Request library to invoke a *POST* method that ultimately stores a JSON object to a DynamoDB table (this part happens in the cloud).
  + **GetItem (url, subitem)** - This function has the parameters **url** as a string and and **subitem** as a Python dictionary. The function returns an complete **item** as a Python dictionary. The **subitem** that is passed to the function is a JSON object that contains ***just the keys*** in an **item** in your individual project. The function uses the Request library to invoke a *GET* method that ultimately retrieves a JSON object in a DynamoDB table (this part happens in the cloud).
  + **DeleteItem (url, item)** -This function has the parameters **url** as a string and **item** as a Python dictionary. The **item** is a JSON object that contains ***just the keys*** in an item in your individual project. The function uses the Request library to invoke a *DELETE*method that ultimately removes a JSON object in a DynamoDB table (this part happens in the cloud).
  + **main()** - A glue function that creates a JSON object in Python, calls **PostItem**to save it, then retrieves it using **GetItem**and finally calls **DeleteItem** to remove the item from the database table.

Notes:

* When getting and deleting items from the database table, it is only necessary to have the keys (pk, sk) to identify the item. The rest of the item is not needed.
* Use the URL: "**https://0jhr0r0epf.execute-api.us-east-2.amazonaws.com/itcc2100/items**"
* AWS credentials are **not**needed anywhere as you will be calling an API Gateway URL and not using the Boto3 library.
* All JSON objects must contain a set of **pk** and **sk** keys, and the combination of the 2 must be unique in the database table. For example, all of the following could exist in the DynamoDB table simultaneously because they are unique (sk + pk):
  + sk:1111, pk:2222
  + sk:1111, pk:3333
  + sk:1111, pk:dogtreat
  + sk:2222, pk:dogtreat
  + sk:2222, pk:3333
* To view your items (and everyone else) before delete, issue the AWS CLI command: **aws dynamodb scan --table-name Macomb**
  + If this fails with error make sure your AWS credentials are set correctly (week 01)