Postman API Testing Lab

Postman is a popular testing tool for API testing. Postman allows developers and testers to create and test various HTTP requests and the corresponding API responses. This lab will go over how to set up Postman in your preferred IDE and run tests on an application.

## Getting Started

1. Let’s first download the Postman desktop application.
   1. <https://www.postman.com/downloads/>
2. Select and run the .exe file to install Postman (on Windows)
3. Now, open a command prompt window and type the following:
   1. pip install requests
4. Make sure the web application is downloaded from GitHub and is running via Flask. Make a note of the URL of where you can access the web app. It will generally be something like this: <http://127.0.0.1:5000>

Postman does offer a web application for API testing, however there are limitations on the web application. For our lab, we will be using the desktop application. The desktop application is free; however, Postman does have an Enterprise version that is not free, but we will not be needing that for this lab.

## GET Requests using Python Requests

Let’s first take a look at the HTTP GET requests, and the corresponding API responses.

1. Create a python file in the IDE of your choice. Let’s name this one test\_get\_requests.py
2. Let’s import the necessary dependencies. For this section, we will only need requests.
   1. import requests
3. Now, let’s give the program the proper URL. Assuming that the web application is running on your local machine, use the following code.
   1. response = requests.get(‘<http://127.0.0.1:5000>’)
4. Alright, now let’s print the API response.
   1. print(response.text)
5. Save your file and open a command prompt (make sure you are in the same directory as your script).
6. Run your script.
   1. python test\_get\_requests.py
7. Analyze the output, is it as you expect? You should see a 200 response code followed by a wall of text.

## POST Requests using Python Requests

Now, we can test POST requests and the corresponding API responses.

1. Create another python file in the IDE of your choice. Let’s name this one test\_post\_requests.py
2. Import the necessary dependencies. For this section, we will only need requests.
   1. import requests
3. Now, let’s give the program the proper URL.
   1. response = requests.post(‘<http://127.0.0.1:5000> / post’, data={‘fruit.name’:’Banana’}
4. Print the API response
   1. print(response.text)
5. Save your file and open a command prompt (making sure you are in the same directory as your script).
6. Run your script.
   1. python test\_post\_requests.py
7. Analyze the output, is it as you expect? You should see a 200 response code followed by a wall of text.

## GET Requests using Postman

Postman offers an intuitive and easy-to-use GUI for API testing. Let’s see how this works, and how the response output differs from using Python Requests.

1. Open Postman.
2. Make sure “GET” is selected in the HTTP Method selector.
3. Copy and paste your URL. Assuming the web application is still running on your local machine, the URL should match what you used in your Python scripts.
4. Press “Send” and look at the output body. Notice how you can change the output formatting within the application. Compare this output to the output from the Python Requests, it should be the same.

## POST Requests using Postman

1. Assuming Postman is still open, and the URL is unchanged, make sure to change the HTTP method to “POST”.
2. Underneath the URL bar, click on “Body”. This will allow you to send key-value pairs to the URL. You can also use the “raw” option to type in your data manually. Try a random key-value pair and hit “Send”.
3. Compare the output to the POST request from Python Requests. How does it differ? How is it similar?