Dream Broker Programming Assignment: Instruction for Checking the program

Prepared by: Humayun Rashid

Email: Humayun.rashid@tuni.fi

The application has been developed using python in windows 10. It has been tested with Postman and curl. The test code has also been installed using Python. The server has been deployed to Heroku and testing code has been integrated with Travis CI. Also, the zip file contains all the files that can be run from local server with following instructions. Details instruction has been given below.

Installation:

Python 3.7.2 : https://www.python.org/downloads/release/python-372/

Curl: https://curl.haxx.se/windows/

Postman: https://www.getpostman.com/downloads/

Modules:

Several modules are need to be install to run the code properly. It is strongly recommended to install the libraries using pip.

PIP: https://pip.pypa.io/en/stable/installing/

After importing pip, it is important to add the environment path url. libraries can be installed using pip from command. Install following modules using pip:

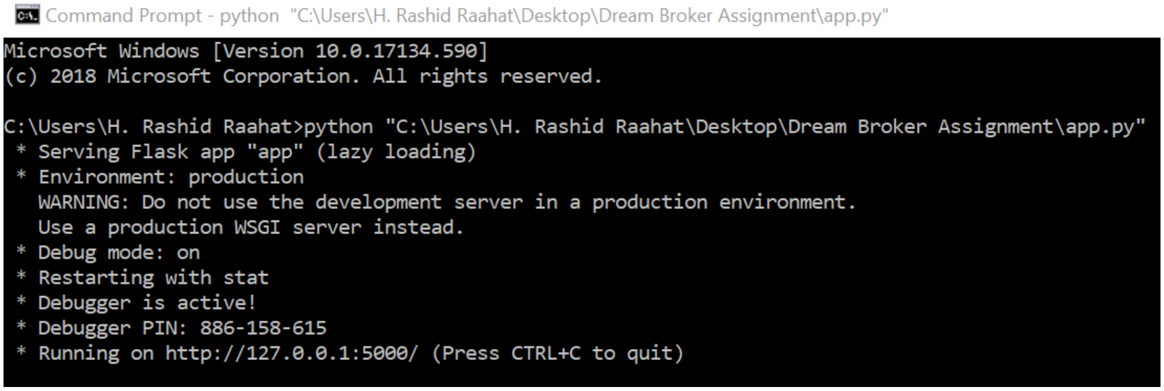
Flask: pip install flask

Requests: pip install requests

unittest : pip install unittest2

Execution:

If Python has been installed properly with run time environment, the code can be executed from command window.


Command Prompt - python "C:\Users\H. Rashid Raahat\Desktop\Dream Broker Assignment\app.py"

```
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\H. Rashid Raahat>python "C:\Users\H. Rashid Raahat\Desktop\Dream Broker Assignment\app.py"
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 886-158-615
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Functionality can be checked through curl command and Postman.

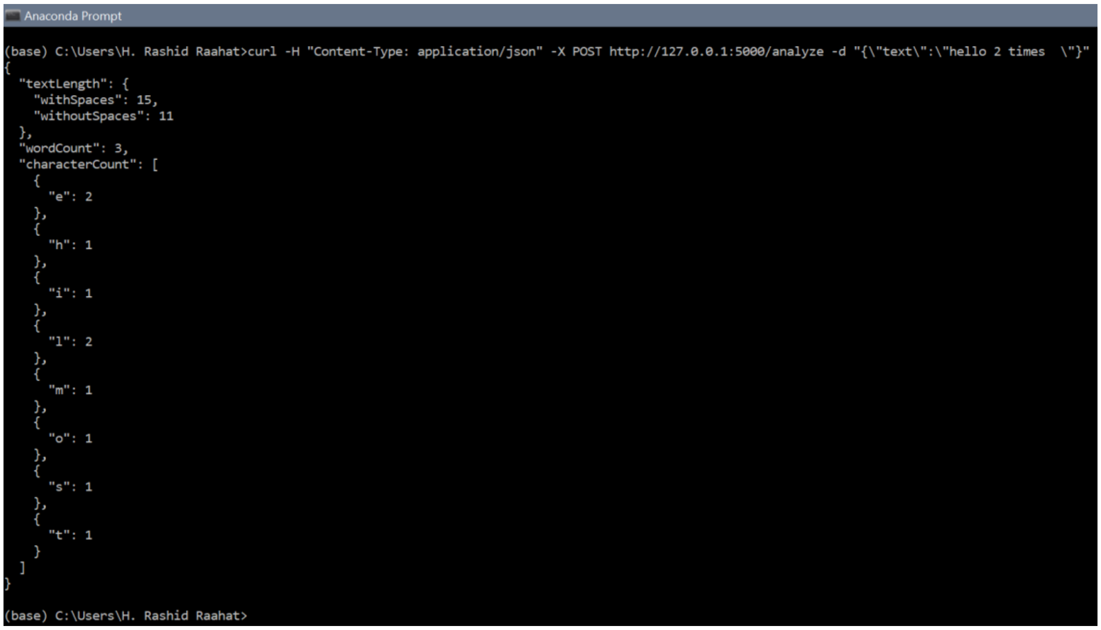
Curl:

It needs to be taken in attention that curl command is different for different OS and different software version. My one is for windows and can be tested with following provided command.

Command: (copy and paste should work if curl is properly installed)

curl -H "Content-Type: application/json" -X POST http://127.0.0.1:5000/analyze -d "{\"text\":\"hello 2 times \"}"

Response:



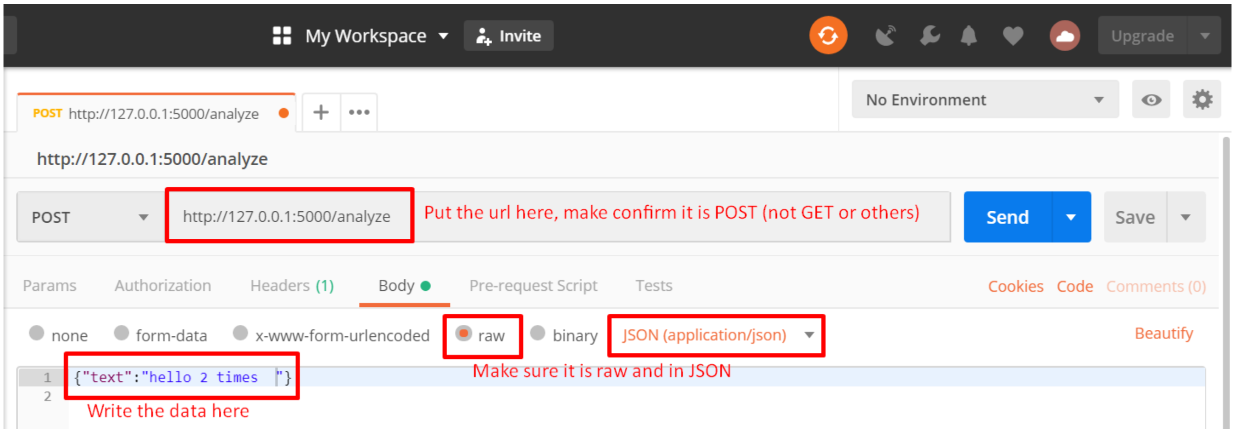
Anaconda Prompt

```
(base) C:\Users\H. Rashid Raahat>curl -H "Content-Type: application/json" -X POST http://127.0.0.1:5000/analyze -d "{\"text\":\"hello 2 times  \"}"
{
  "textLength": {
    "withSpaces": 15,
    "withoutSpaces": 11
  },
  "wordCount": 3,
  "characterCount": [
    {
      "e": 2
    },
    {
      "h": 1
    },
    {
      "i": 1
    },
    {
      "l": 2
    },
    {
      "m": 1
    },
    {
      "o": 1
    },
    {
      "s": 1
    },
    {
      "t": 1
    }
  ]
}

(base) C:\Users\H. Rashid Raahat>
```

Using Postman:

It is very simple to use Postman. Although, this was not asked in the task, but I always prefer to test any API with postman. The setup needs to be like following.



http://127.0.0.1:5000/analyze

POST http://127.0.0.1:5000/analyze

Put the url here, make confirm it is POST (not GET or others)

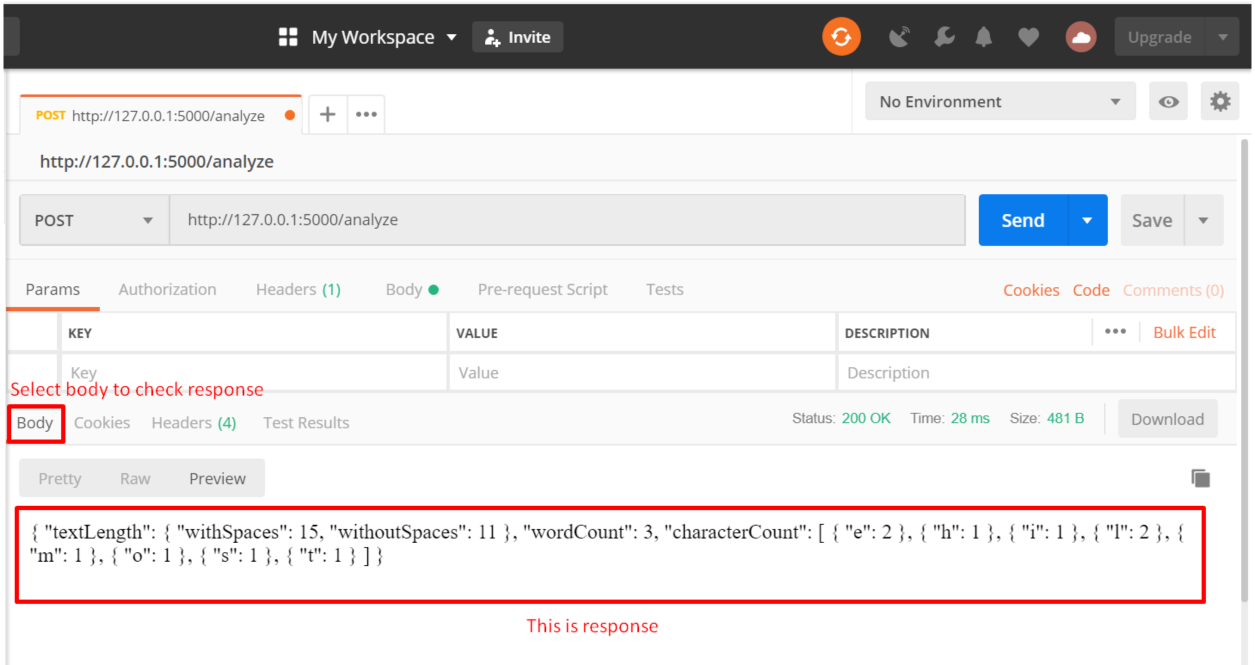
raw JSON (application/json)

Make sure it is raw and in JSON

```
{"text":"hello 2 times  "}
```

Write the data here

Response:



http://127.0.0.1:5000/analyze

POST http://127.0.0.1:5000/analyze

Select body to check response

Status: 200 OK Time: 28 ms Size: 481 B

```
{ "textLength": { "withSpaces": 15, "withoutSpaces": 11 }, "wordCount": 3, "characterCount": [ { "e": 2 }, { "h": 1 }, { "i": 1 }, { "l": 2 }, {
"m": 1 }, { "o": 1 }, { "s": 1 }, { "t": 1 } ] }
```

This is response

Testing:

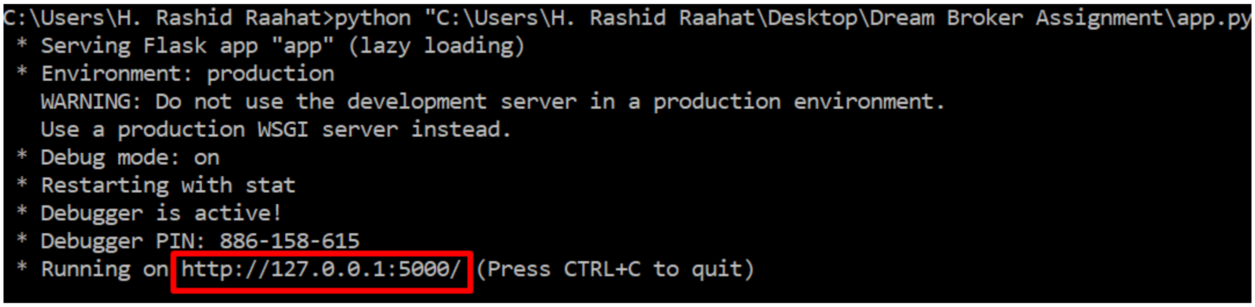
Simple unit testing has been done. The test file can be found in the zip folder. Following this functions/method has been tested:

1. If post request returns the expected result
2. Text length counting test
3. Character counting test
4. Word counting test

For checking, server URL need to be replaced carefully. For my case, server url is: http://127.0.0.1:5000

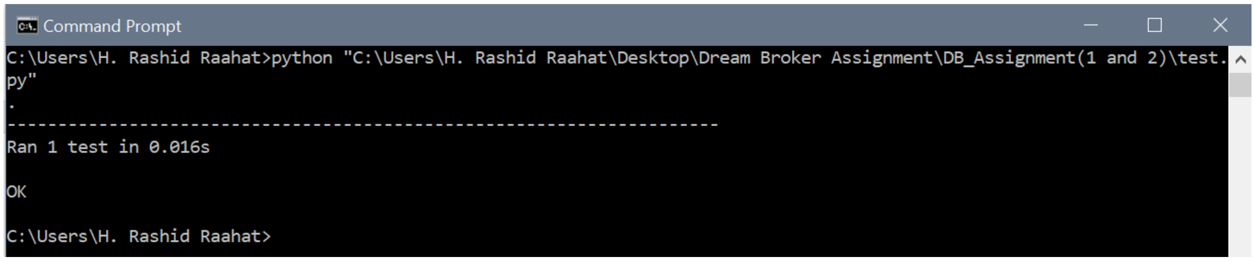
so, in test.py, I used : http://127.0.0.1:5000/analyze in a varable server_url.

For testing, it needs to be replaced with proper localhost url. (Example: http://localhost:8000/analyze)



```
C:\Users\H. Rashid Raahat>python "C:\Users\H. Rashid Raahat\Desktop\Dream Broker Assignment\app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 886-158-615
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

It can be run from command if Python and run time environment has been properly configured. If testcase passes, output will be similar like this:



```
C:\Users\H. Rashid Raahat>python "C:\Users\H. Rashid Raahat\Desktop\Dream Broker Assignment\DB_Assignment(1 and 2)\test.
py"
.
----------------------------------------------------------------------
Ran 1 test in 0.016s

OK

C:\Users\H. Rashid Raahat>
```