In [ ]:
```
callable
|
Decorator
|
Flask - Micro Framework
```

In [2]:
```
class cname:
    def __init__(self):
        print('Welcome')
    def method1(self):
        print('non-constructor')

obj = cname()
obj()
```

```
Welcome
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[2], line 8
      5         print('non-constructor')
      7 obj = cname()
----> 8 obj()

TypeError: 'cname' object is not callable
```

In [3]:
```
callable(obj)
```

Out[3]:
```
False
```

In [4]:
```
def fx():
    pass
callable(fx)
```

Out[4]:
```
True
```

In [6]:
```
print(type(fx))
```

```
<class 'function'>
```

In [7]:
```
class box:
    pass
# box  Vs  box()
obj = box() => obj() Vs fx()
__call__ <== specialmethod
```

```
<class 'type'>
```

In [9]:
```
def f1(*a):
    print('Hello')
    print(a)

f1.__call__() # f1()
f1.__call__(10,20,30) # f1(10,20,30)
```

```
Hello
()
Hello
(10, 20, 30)
```

In [10]:
```
class box:
    def __init__(self,bid,bname):
        self.bid = bid
        self.bname = bname
    def __call__(self):
        return self.bid,self.bname # more than one 1 returns ->tuple
```

In [12]:
```
obj = box(101,'Box-1')
callable(obj)
obj() ## not using obj.<methodName>
```

Out[12]:
```
(101, 'Box-1')
```

In [15]:
```
if(callable(obj)):
    print(obj())
```

```
(101, 'Box-1')
```

In [16]:
```
# device name,id,configuration - use constructor
# use __call__ - get device configuration details
# each device - object

#Router1    Switch  Firewall  Router2

class device_config:
    '''this is device configuration attribute details'''
    def __init__(self,device_name,device_id,device_code=0):
        self.device_name = device_name
        self.device_id = device_id
        self.device_code = device_code
        print(f'Device {self.device_name} configuration is done')
    def __call__(self):
        return self.device_name,self.device_id,self.device_code
```

In [17]:
```
obj1 = device_config('switch','s123',4590)
obj2 = device_config('router','r4902',5902)
obj3 = device_config('switch','s459',5944)
```

```
Device switch configuration is done
Device router configuration is done
Device switch configuration is done
```

In [18]:
```
obj1()
```

Out[18]:
```
('switch', 's123', 4590)
```

In [19]:
```
obj2()
```

Out[19]:
```
('router', 'r4902', 5902)
```

In [20]:
```
obj3()
```

Out[20]:
```
('switch', 's459', 5944)
```

In [ ]:
```
python --->App1 -> V1.0 ----->R+
            |       ||
           App2 -->V1.1 ----->R+...
--------------------------------------//meta programming - Higher code
```

```
adding new features to an existing code without changing the existing code

[Home] [AboutUs]  [News]      [ Contactus ]
                    |-->City1
                    |-->City2

After 3 months
[Home] [AboutUs]  [News]      [ Contactus ]
                    |-->City1
                    |-->City2
                    |==>City3
                    |==>City4


[Home] [AboutUs]  [News]       [Blog ] [ Contactus ]
                    |-->City1
                    |-->City2
                    |==>City3
                    |===>City4
```

In [22]:
```
def f1():
    def f2():
        def App1():
            print('App1 - feature')
        def App2():
            print('App2 - feature')
        def App3():
            print('App3 - feature')
        App1()
        App2()
        App3()
    return f2

r = f1()
r()
```

```
App1 - feature
App2 - feature
App3 - feature
```

In [24]:
```
def f1(a):
    def f2():
        def App1():
            print('App1 - feature')
        def App2():
            print('App2 - feature')
        def App3():
            print('App3 - feature')
        App1()
        App2()
        a() # calling App4 function
        App3()
    return f2

def App4():
    print('App4 - feature')

r = f1(App4)
r()
```

```
App1 - feature
App2 - feature
App4 - feature
App3 - feature
```

In [26]:
```
def f1(a):
    def f2():
        a()
    return f2

def city1():
    print('<h1> City1 news page</h1>')

city1 = f1(city1)
city1()
```

```
<h1> City1 news page</h1>
```

In [27]:
```
def city2():
    print('City2 news page')

city2 = f1(city2)
city2()
```

```
City2 news page
```

In [ ]:
```
def decorator_function(<callable_argument>):
    def function_wrapper(*args,**kwargs):
        <callable_argument>() # callableobject
    return function_wrapper

def myapp():
    ..

#myapp = decorator_function(myapp)
#myapp()

@decorator_function
def myapp():
    ...
    ...
myapp()
```

In [28]:
```
def f1(a1):
    def f2():
        a1()
    return f2

@f1
def myapp1():
    print('this is testApp - myapp1')

@f1
def myapp2():
    print('This is demoApp - myapp2')
```

In [29]:
```
myapp1()
```

```
this is testApp - myapp1
```

In [30]:
```
myapp2()
```

```
This is demoApp - myapp2
```

In [31]:
```
class product:
    product_id = 101
    product_name = 'pA'
```

In [33]:
```
print(product.product_id,product.product_name)
product.product_id = 450 # using class name we can modify an existing attribute
print(product.product_id,product.product_name)
product.product_cost = 1000 # we can add new attribute
print(product.product_cost,product.product_name)
```

```
101 pA
450 pA
1000 pA
```

In [37]:
```
class product:
    product_id = 101
    product_name = 'pA'
    @classmethod
    def f1(cls):
        print(cls.product_id,cls.product_name)
    @classmethod
    def f2(cls,product_id):
        cls.product_id = product_id

product.f1() # f1(product)  Vs  obj.f1() ->f1(obj) ==>def f1(self):
product.f2(505)
product.f1()
```

```
101 pA
505 pA
```

In [42]:
```
class Enrollment:
    @classmethod
    def f1(cls,name,dob,city):
        cls.Name = name
        cls.DOB = dob
        cls.City = city
    def f2(self):
        print(f'About {self.Name} details:-')
        print(f'Emp name:{self.Name} DOB:{self.DOB} Working City:{self.City}')

eobj1 = Enrollment()
Enrollment.f1('','','') # we call only one time - className.<attirbute> = value
eobj1.f2()

eobj2 = Enrollment()
eobj2.f2()
```

```
About  details:-
Emp name: DOB: Working City:
About  details:-
Emp name: DOB: Working City:
```

In [44]:
```
class Enrollment:
    @classmethod
    def f1(cls,name,dob,city):
        cls.Name = name
        cls.DOB = dob
        cls.City = city
    def initialization(self,name,dob,city):
        self.Name = name
        self.DOB = dob
        self.City = city
    def f2(self):
        print(f'About {self.Name} details:-')
        print(f'Emp name:{self.Name} DOB:{self.DOB} Working City:{self.City}')

eobj1 = Enrollment()
Enrollment.f1('','','') # we call only one time - className.<attirbute> = value
```

In [45]:
```
eobj1.f2()
```

```
About  details:-
Emp name: DOB: Working City:
```

In [46]:
```
eobj1.initialization('Arun','1st Jan','City-1')
```

In [47]:
```
eobj1.f2()
```

```
About Arun details:-
Emp name:Arun DOB:1st Jan Working City:City-1
```

In [48]:
```
eobj2 = Enrollment()
eobj2.initialization('Vijay','2nd Feb','City-2')
eobj2.f2()
```

```
About Vijay details:-
Emp name:Vijay DOB:2nd Feb Working City:City-2
```

In [49]:
```
eobj3 = Enrollment()
eobj3.f2()
```

```
About  details:-
Emp name: DOB: Working City:
```

In [50]:
```
class box:
    box_id = 101
    box_name = 'Box-demo'
    @classmethod
    def f1(cls):
        print('Class Method:',cls.box_id,cls.box_name)
    def f2(self):
        print('Object Method:',self.box_id,self.box_name)
    @staticmethod
    def f3():
        '''this is static method - won't access class attributes'''
        print('system info details')

box.f3() # we can invoke static method using class (or) class instance
obj = box()
obj.f3() # we can invoke using object based
```

```
system info details
system info details
```

In [52]:
```
def myf1(arg):
    def wrapper_code(*args,**kwargs):
        result = arg(*args,**kwargs)
        print('arg - function - func will get invoked')
        print('After function runs')
        return result
    return wrapper_code

@myf1
def calc(a,b):
    return a+b

calc(10,20)
```

```
arg - function - func will get invoked
After function runs
```

Out[52]:

```
30
```

In [ ]:

```
@classmethod
@staticmethod
@property
---------------//builtin decorators
```

In [53]:

```
#help(property)
@property decorator is built-in
use attrubutes - without calling then with ( )
used for getter,setting,deleter
```

```
Help on class property in module builtins:

class property(object)
 |  property(fget=None, fset=None, fdel=None, doc=None)
 |
 |  Property attribute.
 |
 |    fget
 |      function to be used for getting an attribute value
 |    fset
 |      function to be used for setting an attribute value
 |    fdel
 |      function to be used for del'ing an attribute
 |    doc
 |      docstring
 |
 |  Typical use is to define a managed attribute x:
 |
 |  class C(object):
 |      def getx(self): return self._x
 |      def setx(self, value): self._x = value
 |      def delx(self): del self._x
 |      x = property(getx, setx, delx, "I'm the 'x' property.")
 |
 |  Decorators make defining new properties or modifying existing ones easy:
 |
 |  class C(object):
 |      @property
 |      def x(self):
 |          "I am the 'x' property."
 |          return self._x
 |      @x.setter
 |      def x(self, value):
 |          self._x = value
 |      @x.deleter
 |      def x(self):
 |          del self._x
 |
 |  Methods defined here:
 |
 |  __delete__(self, instance, /)
 |      Delete an attribute of instance.
 |
 |  __get__(self, instance, owner=None, /)
 |      Return an attribute of instance, which is of type owner.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __set__(self, instance, value, /)
 |      Set an attribute of instance to value.
 |
 |  __set_name__(self, owner, name, /)
 |      Method to set name of a property.
 |
 |  deleter(self, object, /)
 |      Descriptor to obtain a copy of the property with a different deleter.
 |
 |  getter(self, object, /)
 |      Descriptor to obtain a copy of the property with a different getter.
 |
 |  setter(self, object, /)
 |      Descriptor to obtain a copy of the property with a different setter.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __isabstractmethod__
 |
 |  fdel
 |
 |  fget
 |
 |  fset
```

In [57]:
```
class product:
    def __init__(self,pname):
        self._name = pname
    @property
    def display(self):
        return self._name
obj = product('pA')
print(obj._name)
```

```
pA
```

In [62]:
```
class product:
    def __init__(self,pname):
        self._name = pname
    @property
    def display(self): # getter
        print('Getter block')
        return self._name
    @display.setter
    def name(self,pname): # setter
        print('Setter block')
        self._name = pname

obj = product('pA')
print(obj._name)
obj._name = 'productB'
print(obj._name)
obj._name = 'productC'
print(obj._name)
```

```
pA
productB
productC
```

In [ ]:
```
Common Gateway Interface (CGI)
--------------------------------
```

```
 -> Web Concepts
 -> Client <--> Server
 |
 -> Developer = Code + admin
                     |->Install webserver + Configure webserver + start webserv
                     |->Install DataBase + Configure DB + start db daemon(R+)
   -> /var/www/html/<html-Files>
   -> /var/www/cgi-bin/<serverCode>
 ############################  Vs #########################
 Web Framework (or) Framework - Collection of libraries
                                         |->web,db,thread,module etc.,
     - default webserver
     - default database ...
Install WebFramework ->Code
                      ------
 Project_Folder/
                serverCode.py
                templates/ <== pre-defined directory/folder
                     |->login.html
                     |->index.html
                       ..
 =========================================================================
Flask - Web Application Framework written in Python
         WSGI - Web Server Gateway Interface (WSGI)
         Jinja2 template
         ..
        Model View Template (MVT)

open a broswer -> on the addressbar ->Enter your IP ->|    | => IP/page | =>...
--------------------------------------------------------------------------------
requests.get() ->webPage / data(json)
.....................................

import flask <== module
flask.Flask <== className - follows the Constructor - current module  __main__ <
--------------
 |->obj =>Application_object

design style is decorator =>  @Application_object.route(<URL>)
                              def functionName():
                                   ....... response Content

if __name__ == '__main__':
    Application_object.run()
                            debug=True

C:\Users\karth>mkdir FlaskApp

C:\Users\karth>cd FlaskApp

C:\Users\karth\FlaskApp>pip install virtualenv # module installation

C:\Users\karth\FlaskApp>python -m virtualenv myapp1 # create new virtual env
C:\Users\karth\FlaskApp>myapp1/Scripts/activate
(myapp1) C:\Users\karth\FlaskApp>
(myapp1) C:\Users\karth\FlaskApp>

On Windows
=============
C:\Users\karth\FlaskApp>myapp1/Scripts/activate
```

```
(myapp1) C:\Users\karth\FlaskApp>


On Linux
==========
myapp/bin/activate
(myapp1) root@hostname~]#


(myapp1) C:\Users\karth\FlaskApp>

(myapp1) C:\Users\karth\FlaskApp>pip install flask
Collecting flask
  Downloading flask-3.1.2-py3-none-any.whl.metadata (3.2 kB)
Collecting blinker>=1.9.0 (from flask)
  Using cached blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting click>=8.1.3 (from flask)
  Using cached click-8.2.1-py3-none-any.whl.metadata (2.5 kB)
Collecting itsdangerous>=2.2.0 (from flask)
  Using cached itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting jinja2>=3.1.2 (from flask)
  Using cached jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting markupsafe>=2.1.1 (from flask)
  Using cached MarkupSafe-3.0.2-cp310-cp310-win_amd64.whl.metadata (4.1 kB)
Collecting werkzeug>=3.1.0 (from flask)
  Using cached werkzeug-3.1.3-py3-none-any.whl.metadata (3.7 kB)
Collecting colorama (from click>=8.1.3->flask)
  Using cached colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Downloading flask-3.1.2-py3-none-any.whl (103 kB)
Using cached blinker-1.9.0-py3-none-any.whl (8.5 kB)
Using cached click-8.2.1-py3-none-any.whl (102 kB)
Using cached itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Using cached jinja2-3.1.6-py3-none-any.whl (134 kB)
Using cached MarkupSafe-3.0.2-cp310-cp310-win_amd64.whl (15 kB)
Using cached werkzeug-3.1.3-py3-none-any.whl (224 kB)
Using cached colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: markupsafe, itsdangerous, colorama, blinker, werk
Successfully installed blinker-1.9.0 click-8.2.1 colorama-0.4.6 flask-3.1.2 itsd

(myapp1) C:\Users\karth\FlaskApp>

-----------------------------------------------------------------------
1. Open your command prompt
2. On Linux/mac =>  pip3 install virtualenv
   On Winx => python -m pip install virtualenv

3. Create virtual env
C:\Users\karth\FlaskApp>python -m virtualenv myapp1 # create new virtual env

4. Activate virtual env
C:\Users\karth\FlaskApp>myapp1/Scripts/activate
(myapp1) C:\Users\karth\FlaskApp>

5. Install flask =>  pip install flask (on winx)  pip3 install flask (mac/Linux)
|
6. python{Enter}                                        python3 {Enter}
>>> import flask                                        >>> import flask
>>>                                                     >>>
|
--------------------------------------------------------------------------------
|
C:\Users\karth\FlaskApp>myapp1\Scripts\activate

(myapp1) C:\Users\karth\FlaskApp>
(myapp1) C:\Users\karth\FlaskApp>where python
C:\Users\karth\FlaskApp\myapp1\Scripts\python.exe
C:\Users\karth\AppData\Local\Programs\Python\Python310\python.exe

(myapp1) C:\Users\karth\FlaskApp>
|
from flask import Flask

obj = Flask(__name__)

@obj.route("/") # route url (ex: https://www.google.com  https://www.abc.com)
def f1():
        return "<h2><font color=green>Welcome to Flask App</font></h2>"

if __name__ == '__main__':
        obj.run(debug=True)
--------------------------------------------------------------------------------
```

In [63]:
```
fname = "C:\\Users\\karth\\FlaskApp\\templates\\gpage.html"

import requests
r = requests.get('https://www.google.com')
gpage = r.text

with open(fname,'w') as wobj:
    wobj.write(gpage)
```

In [64]:
```
requests.get('http://127.0.0.1:5000').headers
```

Out[64]:
```
{'Server': 'Werkzeug/3.1.3 Python/3.10.0', 'Date': 'Mon, 15 Sep 2025 06:36:34 G
MT', 'Content-Type': 'text/html; charset=utf-8', 'Content-Length': '85', 'Conne
ction': 'close'}
```

In [65]:
```
requests.get('https://www.google.com').headers
```

Out[65]:
```
{'Date': 'Mon, 15 Sep 2025 06:36:55 GMT', 'Expires': '-1', 'Cache-Control': 'pr
ivate, max-age=0', 'Content-Type': 'text/html; charset=ISO-8859-1', 'Content-Se
curity-Policy-Report-Only': "object-src 'none';base-uri 'self';script-src 'nonc
e-Mvt9k69oy2iAGKl5IfwiYQ' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'unsaf
e-inline' https: http:;report-uri https://csp.withgoogle.com/csp/gws/other-hp",
'Accept-CH': 'Sec-CH-Prefers-Color-Scheme', 'P3P': 'CP="This is not a P3P polic
y! See g.co/p3phelp for more info."', 'Content-Encoding': 'gzip', 'Server': 'gw
s', 'X-XSS-Protection': '0', 'X-Frame-Options': 'SAMEORIGIN', 'Set-Cookie': 'AE
C=AVh_V2jpG9jl-2a08AaFA-XoFjVEBImJJ-skjAS6RsuraG_rtMDtapQ_40Y; expires=Sat, 14-
Mar-2026 06:36:55 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=l
ax, NID=525=ZcLLAWG69sav5o5zp__kAx5uQnoybD_Eiamx9xWWyey0lzPoQ0igrekIEG7II0Svzos
iU4EHqnBPElIYo77Zt7Tc7eYVERl_S4Gr7K2trRjp86gHj7bUMfjUDt6jIDcJpt21-oKfX6tyNwRQtq
GKs3T6YUYHZNZCtXkh3IMGbxaavTbuAMbBEEB5xWXROHMdskNqYbpgM9hh_VcGasQ; expires=Tue,
17-Mar-2026 06:36:55 GMT; path=/; domain=.google.com; HttpOnly', 'Alt-Svc': 'h3
=":443"; ma=2592000,h3-29=":443"; ma=2592000', 'Transfer-Encoding': 'chunked'}
```

In [ ]:
```
jinja2 template Code
----------------------
-> web template - embedded with html tag
```

```
Score: |    | <==    <h2>Score:{{template_variable}}</h2>

{{variable}}

{{Expression}}

{% if condition %}
  {{body}}
{% endif %}

{% for variable in <Collection> %}
   {{variable}}
{% endfor %}

--------------------------------------------------------------------
<UR>/Input <=== To build the URL dynamically - variable
<variableName>

@api.route("/mypage/<user_defined_variable>")
def function(<user_defined_variable>):
    ....
    return ....

------------------------------------------------------------
```

In [ ]:

```
Task
------
|->Create a flask webApp
       |-> /  ->display string
       |-> /aboutus ->display htmlFile

Task
|->import request module
    |->get() - read the above URLs
                           |->get webHeader info
                           |->Content-Type and read a content
                           |->display content to monitor.
------------------------------------------------------------------
```

In [66]:

```
import requests
requests.get('http://localhost:5000')
```

Out[66]:

```
<Response [200]>
```

In [68]:

```
r = requests.get('http://localhost:5000')
r.headers
```

Out[68]:

```
{'Server': 'Werkzeug/3.1.3 Python/3.10.0', 'Date': 'Mon, 15 Sep 2025 08:44:51 G
MT', 'Content-Type': 'text/html; charset=utf-8', 'Content-Length': '12', 'Conne
ction': 'close'}
```

In [69]:

```
r.text
```

Out[69]:

```
'Test message'
```

In [70]:

```
r = requests.get('http://localhost:5000/aboutus')
r.status_code
```

```
Out[70]: 200
```

```
In [71]: r.headers
```

```
Out[71]: {'Server': 'Werkzeug/3.1.3 Python/3.10.0', 'Date': 'Mon, 15 Sep 2025 08:46:32 G
         MT', 'Content-Type': 'text/html; charset=utf-8', 'Content-Length': '125', 'Conn
         ection': 'close'}
```

```
In [72]: r.text
```

```
Out[72]: '<html>\n<head>\n<title> Welcome </title>\n<body>\n<h1> List of available cours
         es </h1>\n<h2> </font> </h2>\n</body>\n</head>\n</html>'
```

requests.get('http://localhost:5000')

```
In [ ]: @app.route('/mypage/<var>') =>   http://localhost:5000/mypage/data 1334  44.23
                                     |__________________________________________|____|______|


        int
        float
        path
        -------//types
        @app.route('/mypage/<int:var>') =>   http://localhost:5000/mypage/data 1334  44.
                                 =======                                         |     |__OK
                                                                                Error

        @app.route('/mypage/<path:fname>') ==> http://localhost:5000/C:\\Demo\\d1\\e1.lo
        -----------------------------------------------------------------------------------
        url_for() - build the URL specific function => url_for(<function>)

        @app.route('/mypage')
        def f1():
            return 'mypage'

        @app.route('/mydept/<dept>')
        def f2(dept):
            return f"working dept is:{dept}"
        ..
        @app.route("/user/<user>")
        def f3(user):
            if(user == 'QA'):
                return redirect(url_for('f1'))
            else:
                return redirect(url_for('f2',dept='Sales'))
                                          ============//keyword argument

        http://localhost:5000/user/QA
        http://localhost:5000/user/prod

        =================================================================

        obj = Flask(__name__)
        @obj.route("/")
        def f1():
            '''return and response web content'''

        @obj.route("/page1")
        def f2():
            '''return page1 repoponse content'''
        @obj.route("/page2/<var>")
```

```
def f3(var):
    '''receive inputVariable from URL and do process and return/response page'''

@obj.route("/page3/<var>")
def f4(var):
    return render_template('reponse_html.html',template_Var = var)

@obj.route("/page4/<var>")
def f5(var):
    if (var == 'patternN'):
        return redirect(url_for('f3',var=<var_value>))
    else:
        return redirect(url_for('f4',var="test message"))
```

In [ ]:
```
HTTP Methods
GET   - Send data - unencrypted format - cached

POST  - Send data - form data - not cached by server

PUT - replace /update the content

DELTE - remove


+---------------------------+
| inputHtml_file.html
|        _______________
| Name:|____arun_________|   (ex: <input type="text" name="n1">
|         _______________
| DOB : |_______________|   (ex: <input type="text" name="n2">)
|         ______________
| City: |_City1______|   (ex: <input type="text" name="n3">)
|
|  (Submit)  (Cancel)
|      |----------------(1)------------------>{'n1':'arun','n2':None,'n3':'City1
+-----------------------------+           ------------------------------------
                                           |
                                           | (2)
                                           |
                                          request.form['n1']  -> 'arun'
                                          request.form['n2']  ->  None
                                          request.form['n3']  -> 'City1'
                                                                  |__(3)
                                                                     |-->te
                                                                         ..
```

In [79]:
```
requests.get('http://localhost:5000/data').headers
```

Out[79]:
```
{'Server': 'Werkzeug/3.1.3 Python/3.10.0', 'Date': 'Mon, 15 Sep 2025 10:35:59 G
MT', 'Content-Type': 'text/html; charset=utf-8', 'Content-Length': '32', 'Conne
ction': 'close'}
```

In [80]:
```
requests.get('http://localhost:5000/data').headers
```

Out[80]:
```
{'Server': 'Werkzeug/3.1.3 Python/3.10.0', 'Date': 'Mon, 15 Sep 2025 10:37:14 G
MT', 'Content-Type': 'application/json', 'Content-Length': '55', 'Connection':
'close'}
```

```
In [ ]:
```