Using requests.get() in Python

Detailed Explanation of Parameters and Usage

Introduction

- Using requests.get() in Python to send GET requests to a URL to retrieve data from the server. The requests library offers several parameters for further customization.

Basic GET Request

Example:

```
import requests
response = requests.get('https://example.com')
```

1. url - Request URL

Description: The URL or endpoint you want to send the request to.

Example:

```
response = requests.get('https://api.example.com/data')
```

2. params - Query Parameters

Description: Adds query parameters to the URL.

Example:

```
params = {'search': 'python', 'page': 2}
response = requests.get('https://api.example.com/search', params=params)
```

Final URL:
https://api.example.com/search?search=python&page=2

3. headers - Request Headers

Description: Adds extra information, like Content-Type or Authorization.

Example:

```
headers = {
    'User-Agent': 'my-app',
     'Authorization': 'Bearer YOUR_ACCESS_TOKEN'
}
response =
requests.get('https://api.example.com/data',
headers=headers)
```

4. auth - Authentication

Description: If the service requires authentication, send username and password.

Example:

```
from requests.auth import HTTPBasicAuth

response =
requests.get('https://api.example.com/secure-data',
auth=HTTPBasicAuth('username', 'password'))
```

5. cookies - Cookies

Description: Sends cookies with the request.

Example:

```
cookies = {'session_id': '123456'}

response = requests.get('https://api.example.com/data',
cookies=cookies)
```

6. timeout - Timeout

Description: Sets max time before ending the request.

Example:

```
response = requests.get('https://api.example.com/data', timeout=5)
```

Or separate connect and read timeouts:

```
response = requests.get('https://api.example.com/data', timeout=(3, 7))
```

7. allow_redirects - Redirects

Description: Specifies whether to follow automatic redirects.

Example:

```
response = requests.get('https://api.example.com/redirect', allow_redirects=False)
```

8. proxies - Proxies

Description: Sends request through a proxy server.

Example:

```
proxies = {
  'http': 'http://10.10.1.10:3128',
  'https': 'https://10.10.1.10:1080'
}
response = requests.get('https://api.example.com/data',
proxies=proxies)
```

9. stream - Streaming

Description: Allows streaming, useful for processing data as it arrives.

Example:

```
response = requests.get('https://api.example.com/large-file', stream=True)
```

To read in chunks:

```
for chunk in response.iter_content(chunk_size=1024):
    print(chunk)
```

10. verify - SSL Verification

Description: Specifies whether to verify SSL certificates.

Example:

```
response = requests.get('https://self-signed.badssl.com/', verify=False)
```

Complete Example with All Parameters

Example:

```
import requests
from requests.auth import HTTPBasicAuth
url = 'https://api.example.com/data'
params = {'search': 'python', 'page': 2}
headers = {'User-Agent': 'my-app'}
cookies = {'session_id': '123456'}
timeout = 5
proxies = {
   'http': 'http://10.10.1.10:3128',
   'https': 'https://10.10.1.10:1080'
}
response = requests.get(url, params=params, headers=headers,
auth=HTTPBasicAuth('username', 'password'), cookies=cookies, timeout=timeout,
allow_redirects=True, proxies=proxies, stream=False, verify=True)
print(response.status_code)
print(response.json())
```