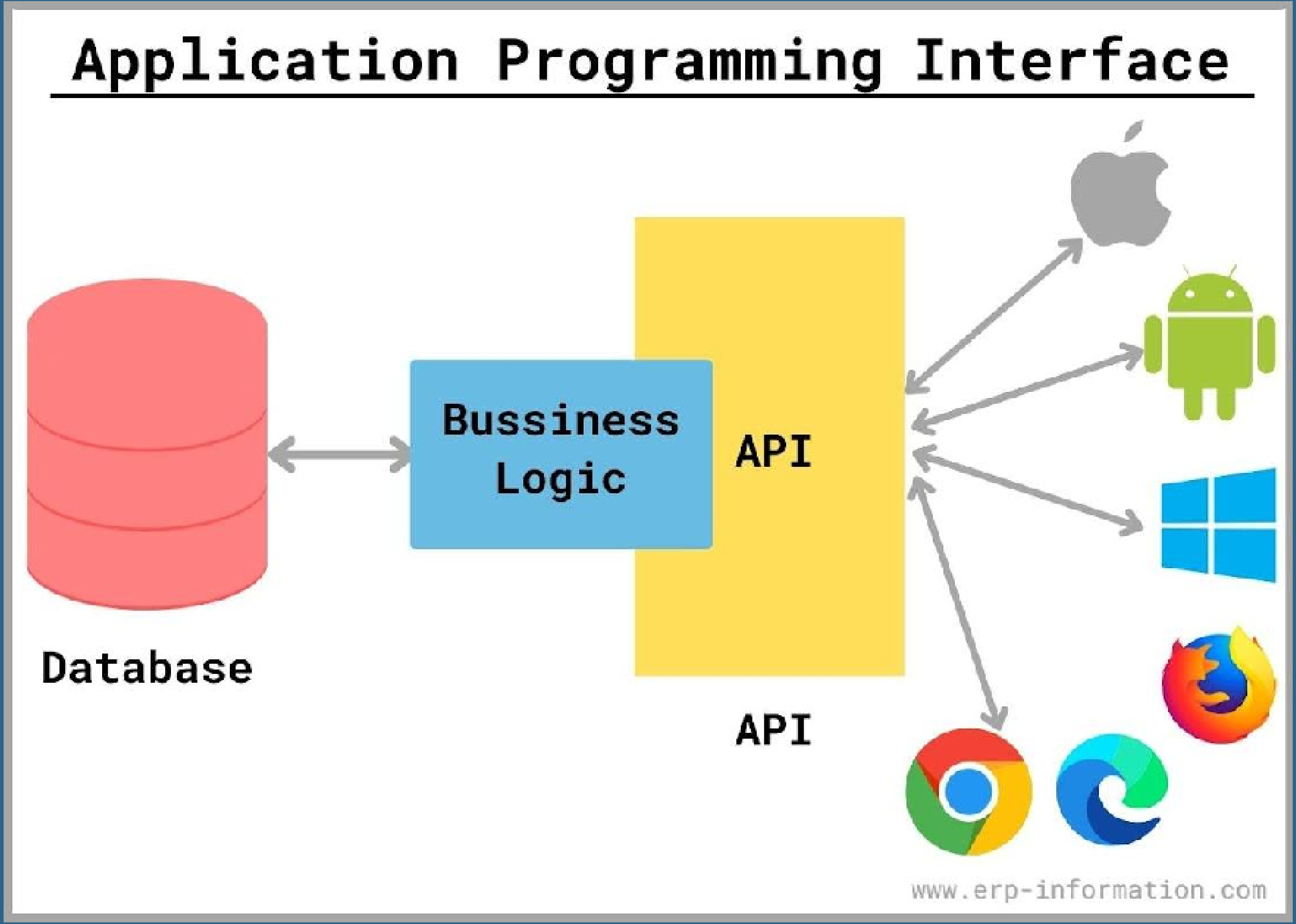
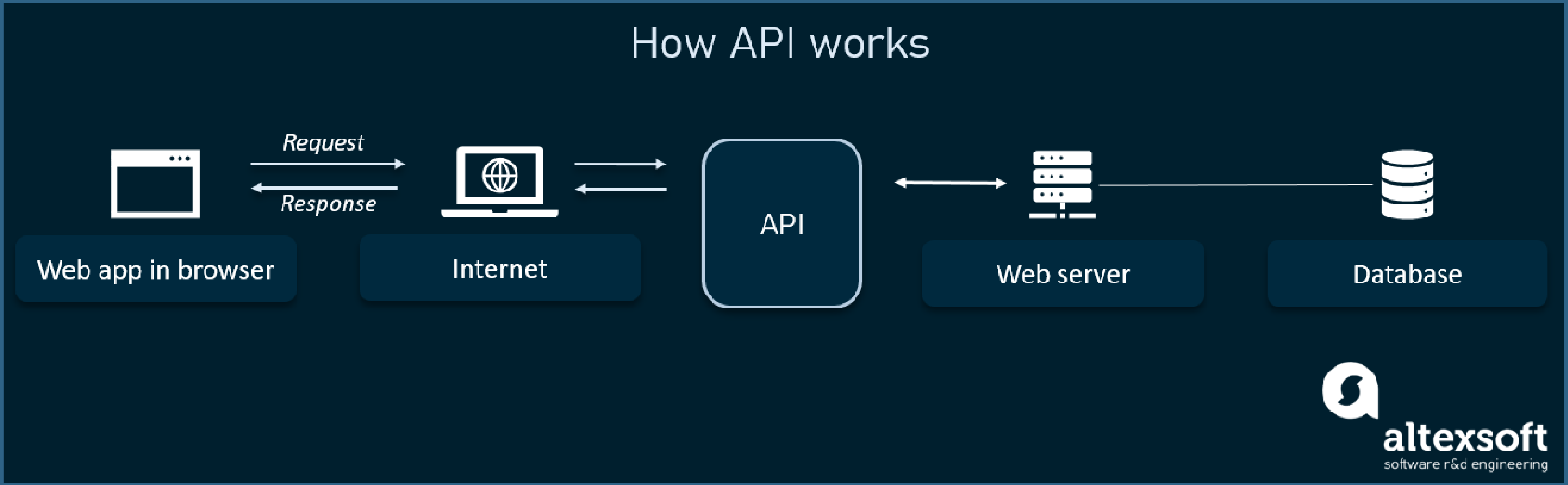
PNDC PROJECT

API (Application Programming Interface)



How does it works ?

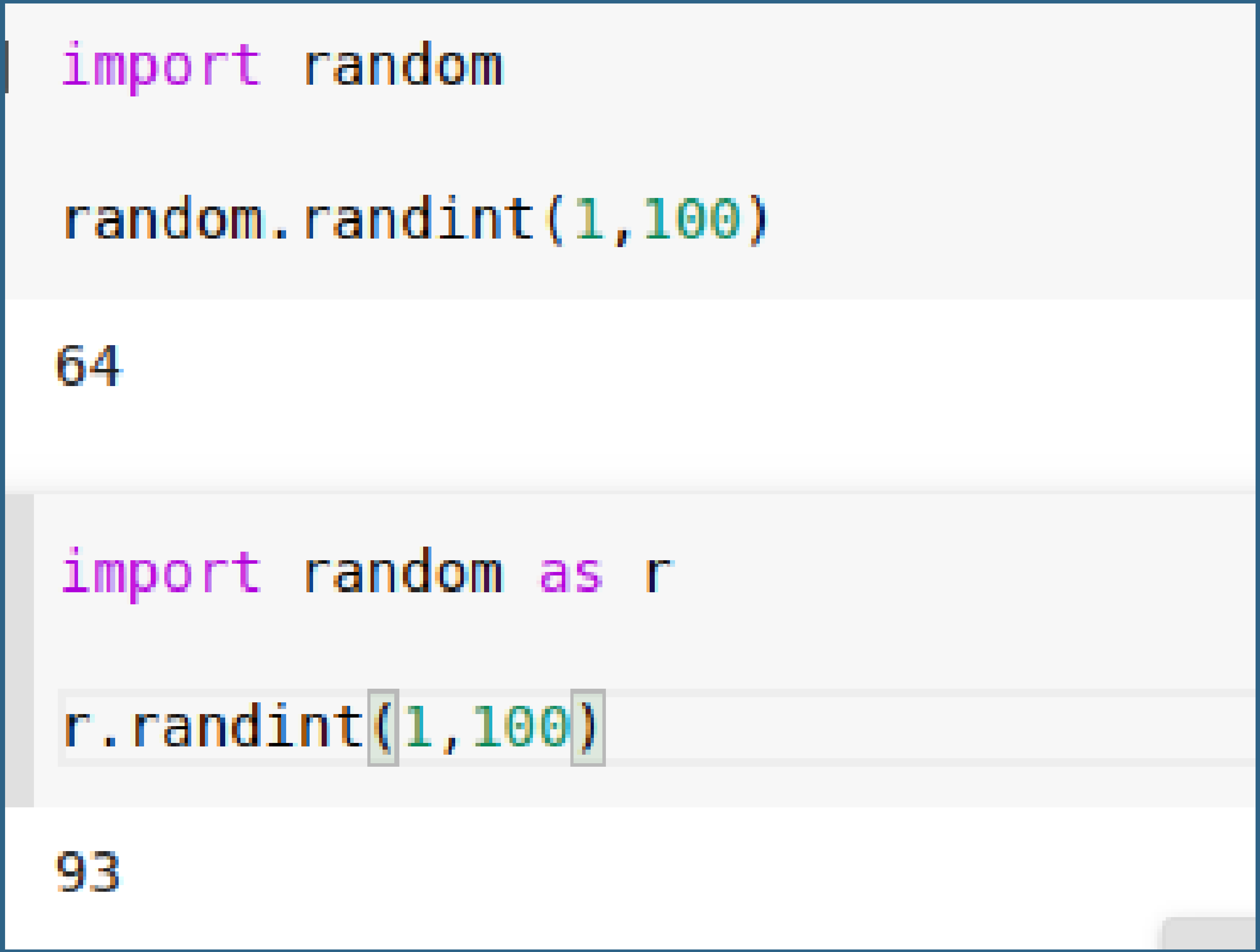


Importing modules in python

```
import module_name
```

```
import module_name as alias
```

Usecase Example



```
import random

random.randint(1,100)
```

```
64
```

```
import random as r

r.randint(1,100)
```

```
93
```

Project Modules

tkinter

It is the standard GUI library for Python & provides a fast and easy way to develop GUI application.

requests

It is a powerful tool that provides the simple elegance of Python to make HTTP requests to any API in the world.

time

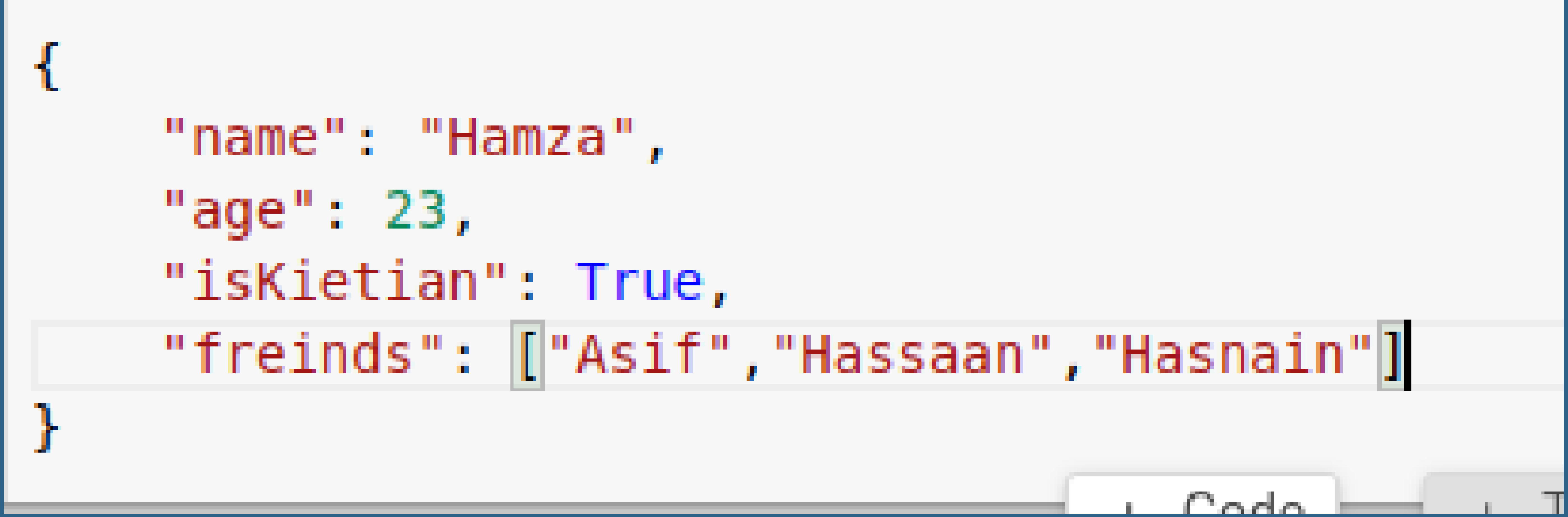
The time module is a standard Python module that contains time access and conversion functions.

JSON (javascript object notation)

JSON (JavaScript Object Notation) is a text-based data exchange format. It is a collection of key-value pairs where the key must be a string type, and the value can be of any of the following types:

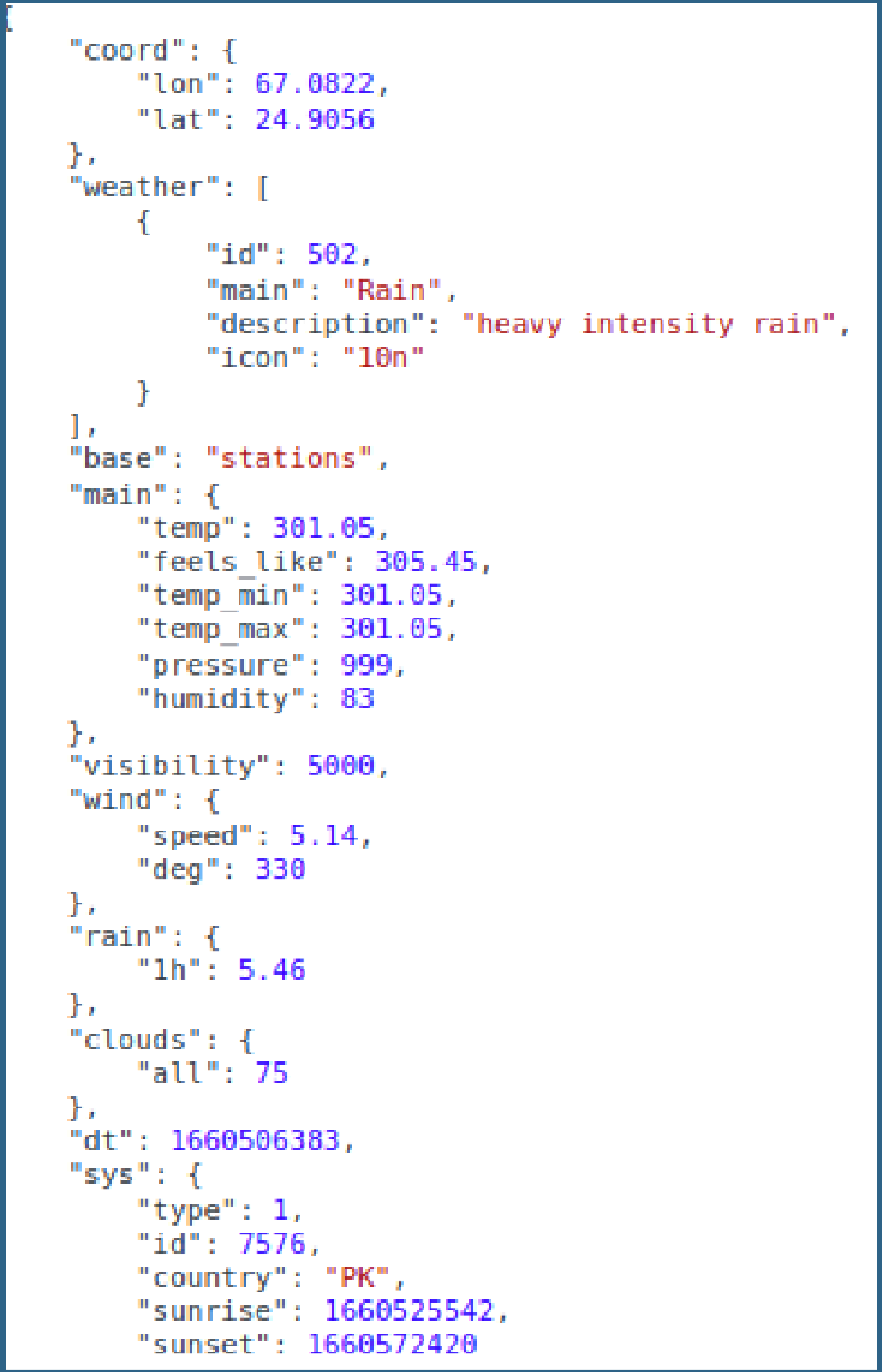
Number , String , Boolean , Array , Object , Null

Simple JSON data

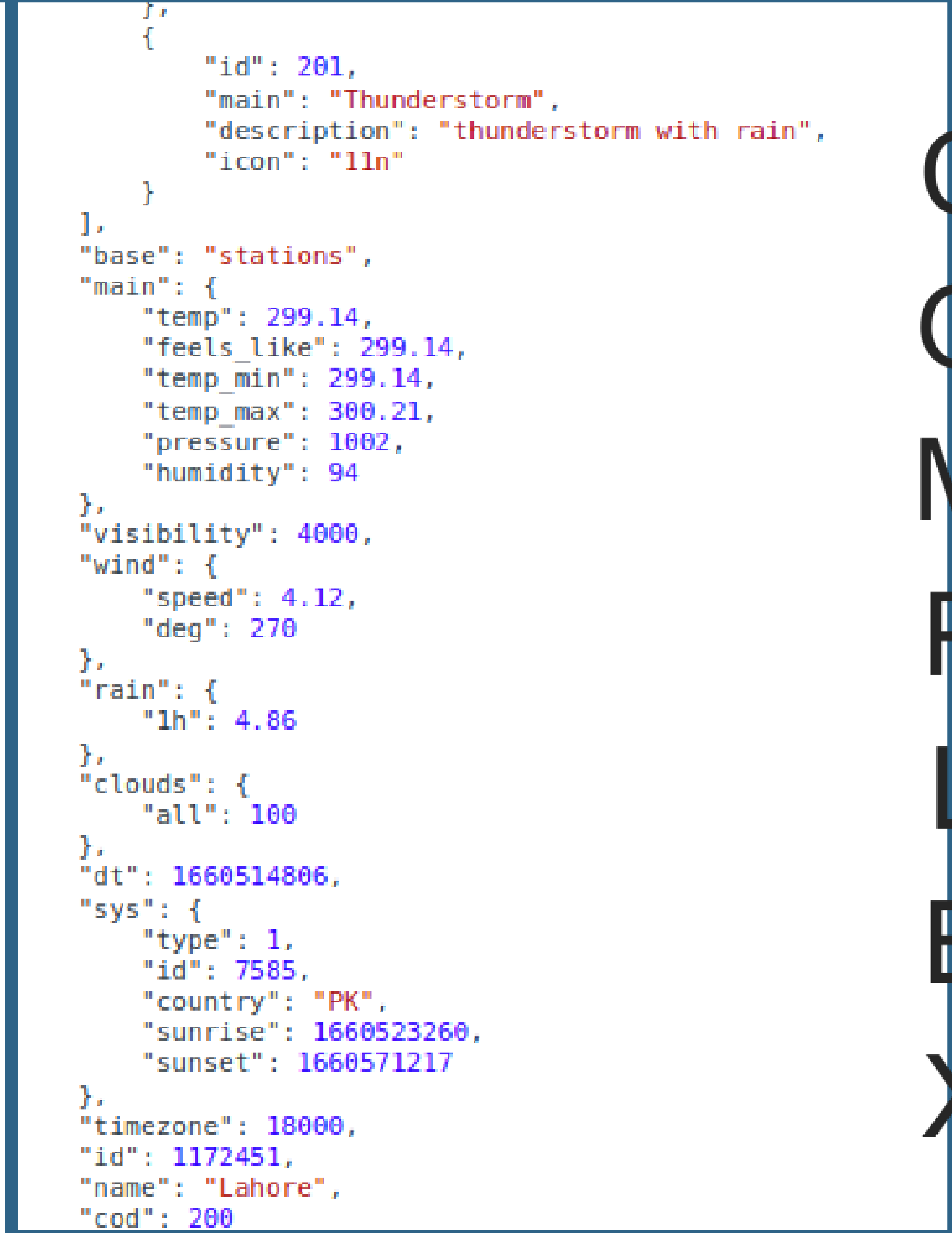


```
{
    "name": "Hamza",
    "age": 23,
    "isKietian": True,
    "freinds": ["Asif","Hassaan","Hasnain"]
}
```

COMPLEX JSON DATA



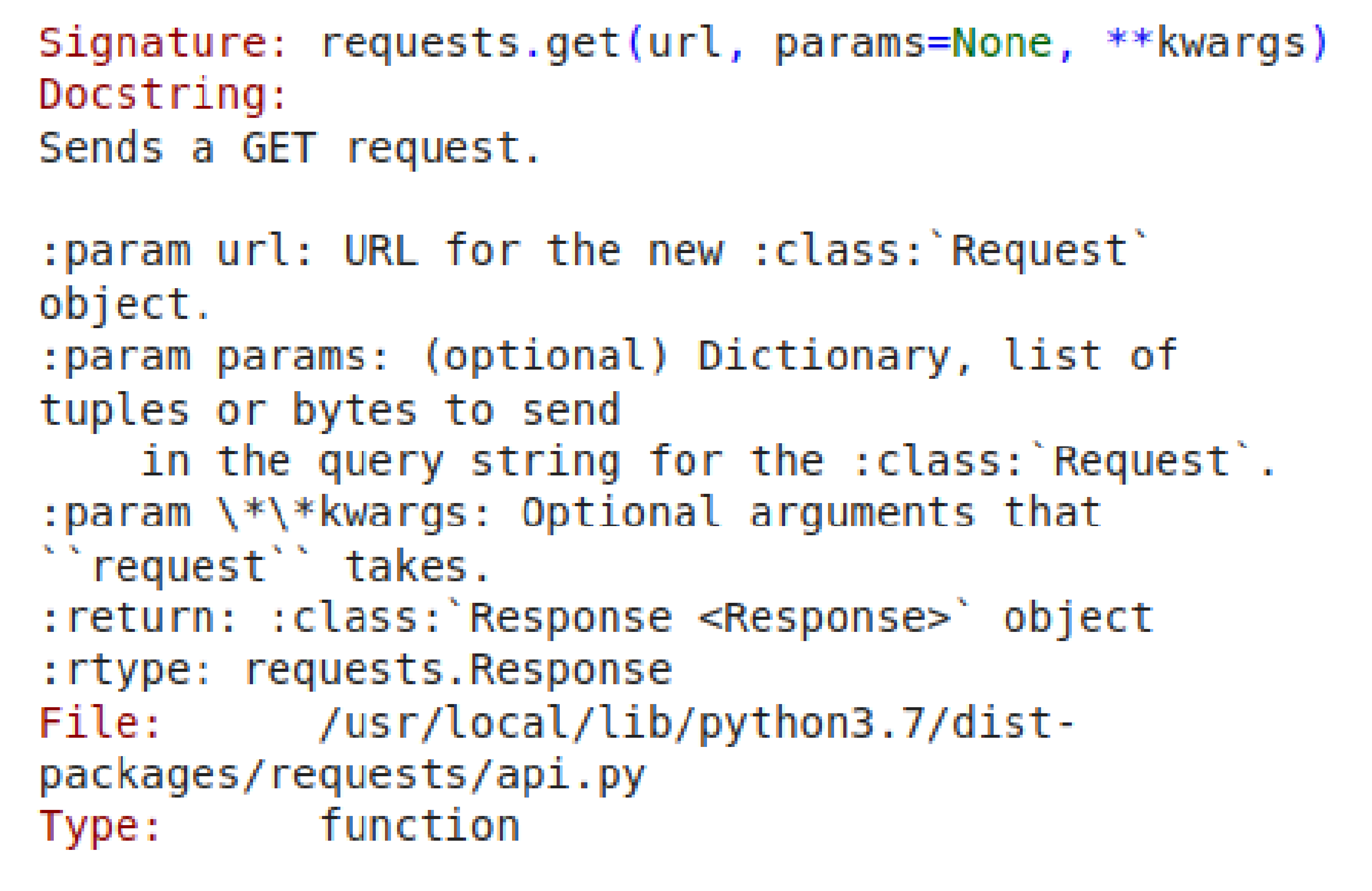
```
{
    "coord": {
        "lon": 67.0822,
        "lat": 24.9056
    },
    "weather": [
        {
            "id": 502,
            "main": "Rain",
            "description": "heavy intensity rain",
            "icon": "10n"
        }
    ],
    "base": "stations",
    "main": {
        "temp": 301.05,
        "feels_like": 305.45,
        "temp_min": 301.05,
        "temp_max": 301.05,
        "pressure": 999,
        "humidity": 83
    },
    "visibility": 5000,
    "wind": {
        "speed": 5.14,
        "deg": 330
    },
    "rain": {
        "1h": 5.46
    },
    "clouds": {
        "all": 75
    },
    "dt": 1660506383,
    "sys": {
        "type": 1,
        "id": 7576,
        "country": "PK",
        "sunrise": 1660525542,
        "sunset": 1660572420
```



```
        },
        {
            "id": 201,
            "main": "Thunderstorm",
            "description": "thunderstorm with rain",
            "icon": "11n"
        }
    ],
    "base": "stations",
    "main": {
        "temp": 299.14,
        "feels_like": 299.14,
        "temp_min": 299.14,
        "temp_max": 300.21,
        "pressure": 1002,
        "humidity": 94
    },
    "visibility": 4000,
    "wind": {
        "speed": 4.12,
        "deg": 270
    },
    "rain": {
        "1h": 4.86
    },
    "clouds": {
        "all": 100
    },
    "dt": 1660514806,
    "sys": {
        "type": 1,
        "id": 7585,
        "country": "PK",
        "sunrise": 1660523260,
        "sunset": 1660571217
    },
    "timezone": 18000,
    "id": 1172451,
    "name": "Lahore",
    "cod": 200
```

request module methods

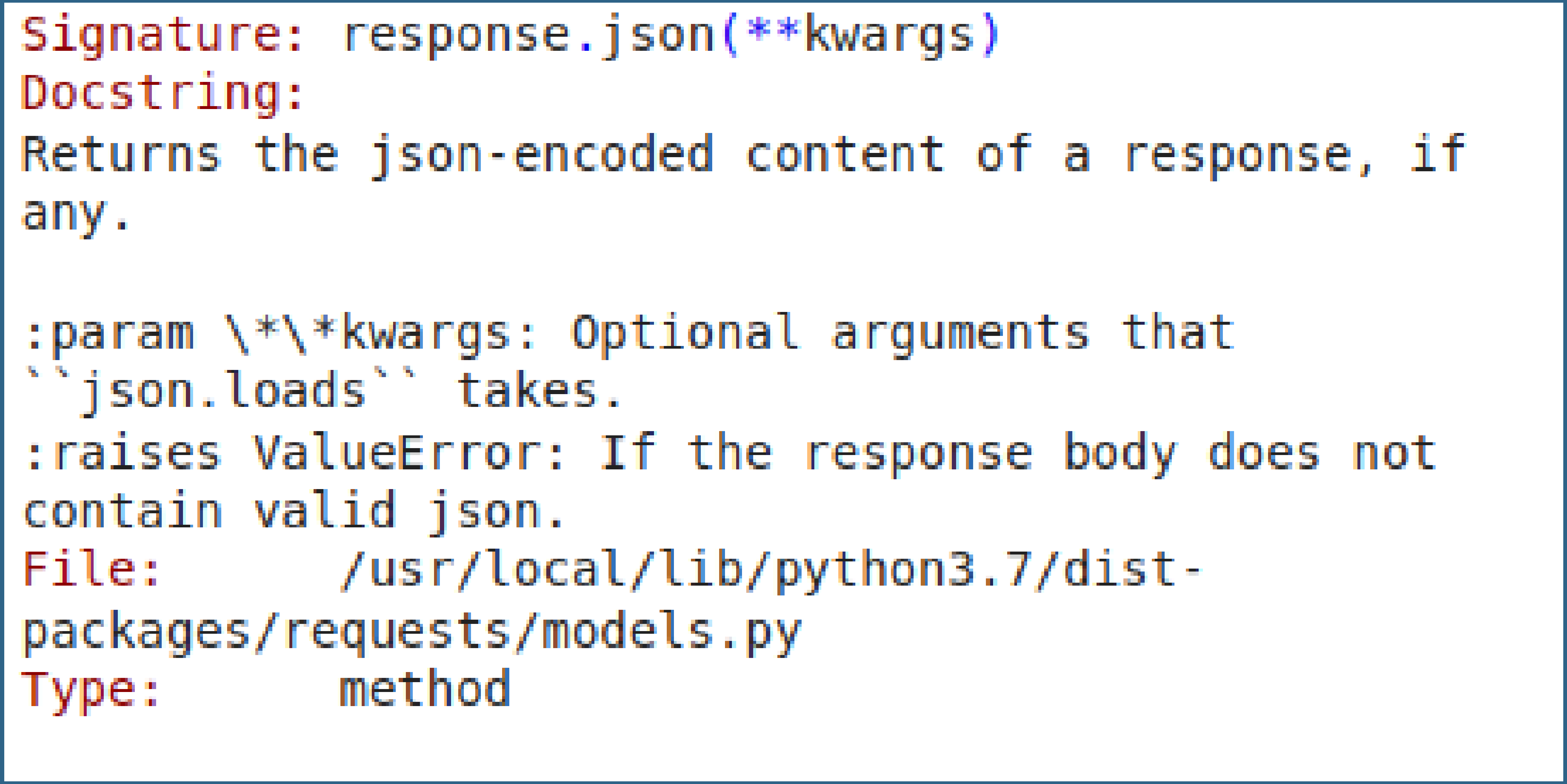
1.



```
Signature: requests.get(url, params=None, **kwargs)
Docstring:
Sends a GET request.

:param url: URL for the new :class:`Request`
object.
:param params: (optional) Dictionary, list of
tuples or bytes to send
    in the query string for the :class:`Request`.
:param \*\*kwargs: Optional arguments that
``request`` takes.
:return: :class:`Response <Response>` object
:rtype: requests.Response
File:      /usr/local/lib/python3.7/dist-
packages/requests/api.py
Type:      function
```

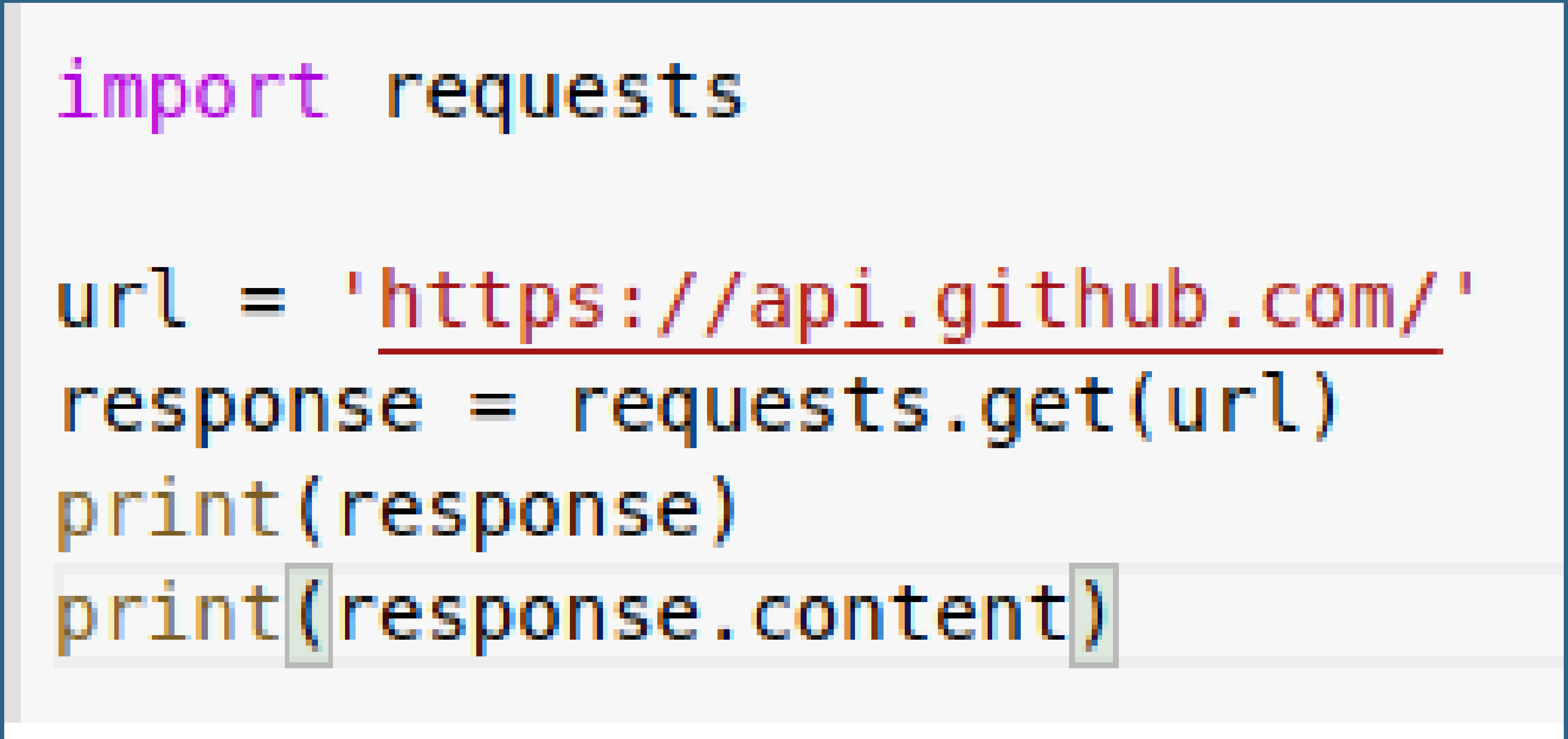
2.



```
Signature: response.json(**kwargs)
Docstring:
Returns the json-encoded content of a response, if
any.

:param \*\*kwargs: Optional arguments that
``json.loads`` takes.
:raises ValueError: If the response body does not
contain valid json.
File:      /usr/local/lib/python3.7/dist-
packages/requests/models.py
Type:      method
```

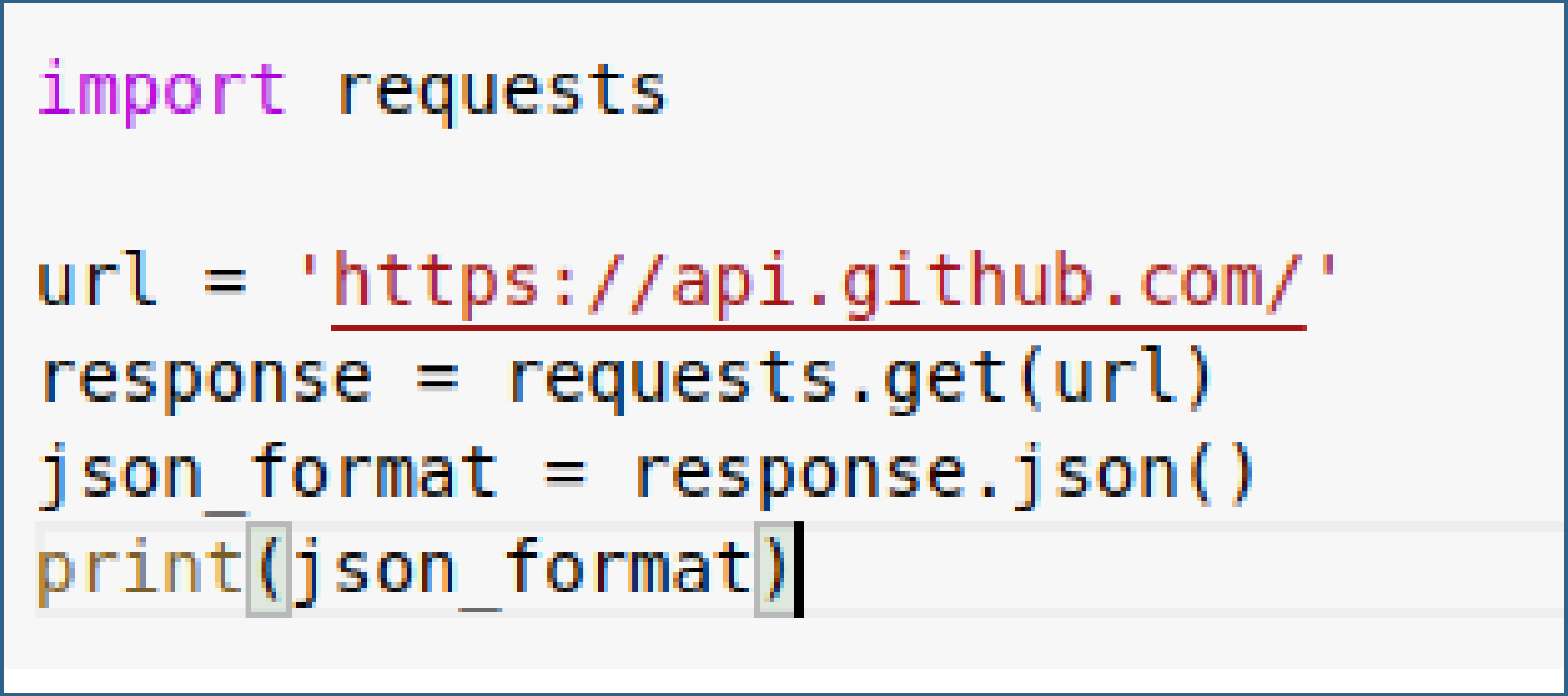
Making a simple request



```python
import requests

url = 'https://api.github.com/'
response = requests.get(url)
print(response)
print(response.content)
```

Converting complex data to JSON format

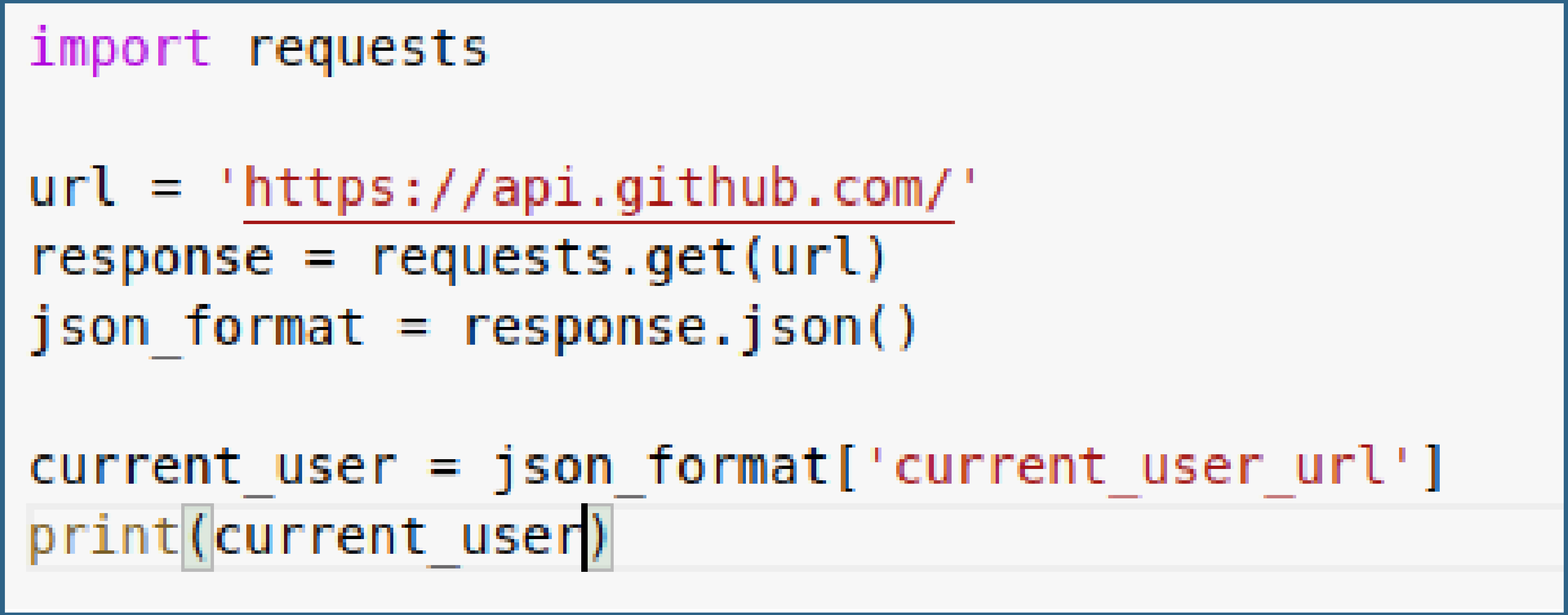


```python
import requests

url = 'https://api.github.com/'
response = requests.get(url)
json_format = response.json()
print(json_format)
```

Acessing JSON Data

json_object ['key']



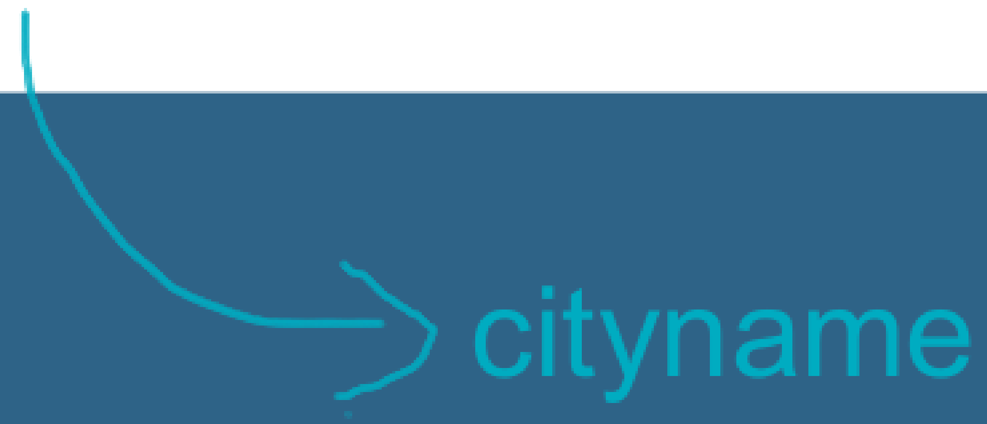
```
import requests

url = 'https://api.github.com/'
response = requests.get(url)
json_format = response.json()

current_user = json_format['current_user_url']
print(current_user)
```

Weather API

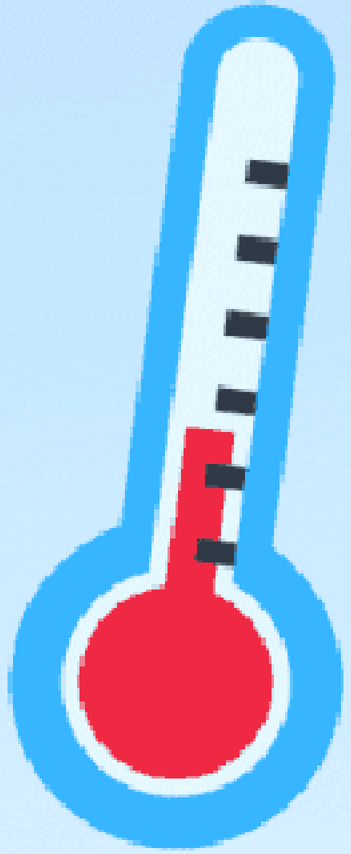
```
"https://api.openweathermap.org/data/2.5/weather?
q="+____+"&appid=f17a8ab707d93da8b2ee85bb36b4fbd3"
```



cityname

How to Convert Kelvin to Celsius

°C = K – 273.15



Convert 300 K to °C:

°C = 300 - 273.15

°C = 26.85 °C

°C = 27 °C (rounded)

Remember, Celsius temperatures have a degree symbol, but Kelvin temperatures do not.

sciencenotes.org

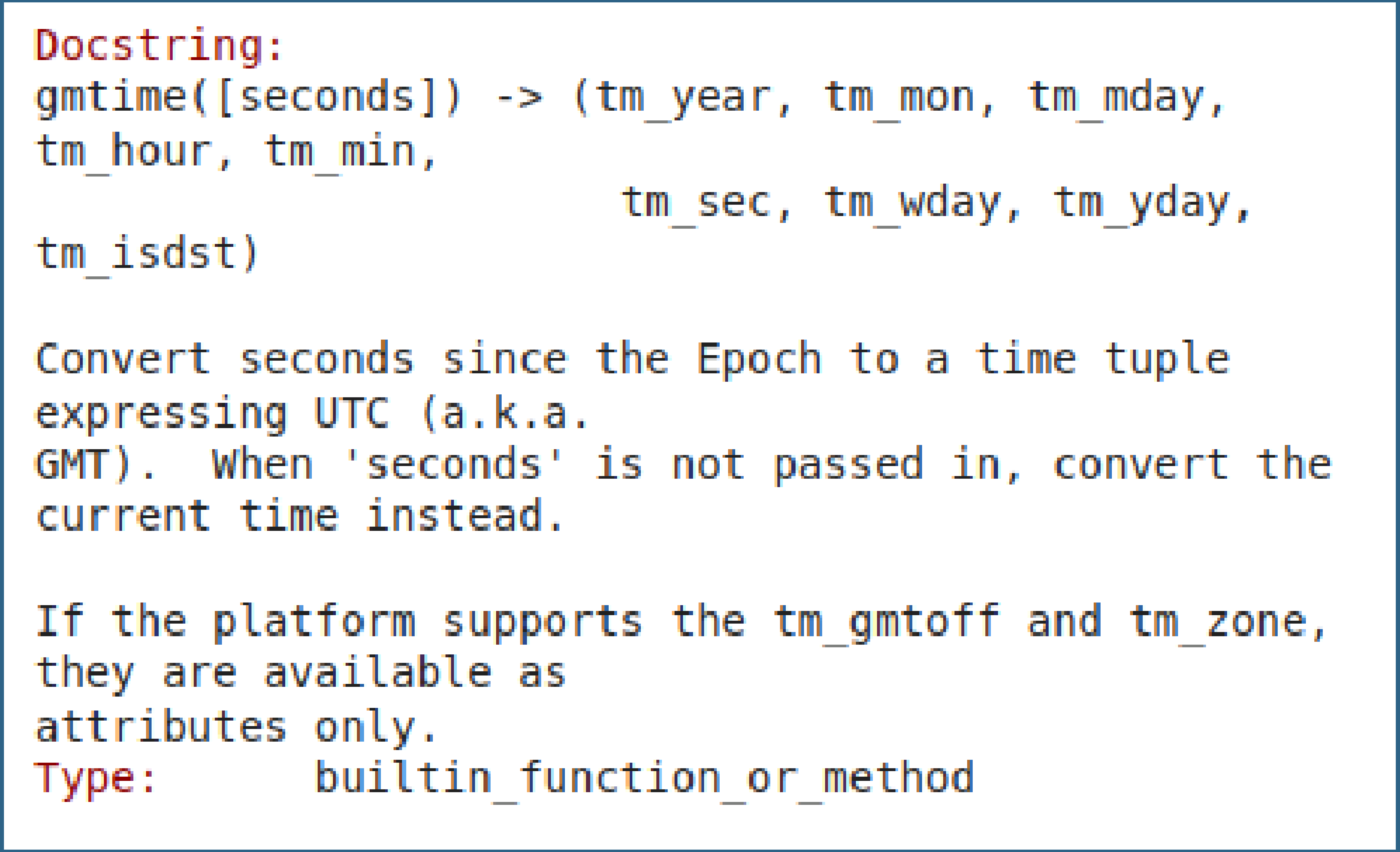
EPOC TIME CONVERTER

https://www.epochconverter.com/

API RESPONSE FIELDS

https://openweathermap.org/current

TIME MODULE METHOD



```
Docstring:
gmtime([seconds]) -> (tm_year, tm_mon, tm_mday,
tm_hour, tm_min,
                       tm_sec, tm_wday, tm_yday,
tm_isdst)

Convert seconds since the Epoch to a time tuple
expressing UTC (a.k.a.
GMT).  When 'seconds' is not passed in, convert the
current time instead.

If the platform supports the tm_gmtoff and tm_zone,
they are available as
attributes only.
Type:      builtin_function_or_method
```

THE END