



# Sanic Web Framework

Lập trình Web trên Python

cuong@techmaster.vn

---

**Tại sao phải lập trình Web  
trên Python trong khi  
Node.js, PHP, Golang đã làm  
rất tốt?**

# Lý do đây



1. Python chuyên để xử lý, phân tích dữ liệu, học máy. Cần phải có giao diện để cho người dùng tương tác. Nếu phải chuyển sang ngôn ngữ khác Python, lập trình sẽ vất vả hơn.
2. Python cũng là một ngôn ngữ tốt để lập trình web
3. Tốc độ Python Web Framework đã được cải thiện nhanh chóng như công nghệ AsyncIO, UVLoop

# Python Web Framework nổi tiếng



- Django: CMS nổi tiếng nhưng chậm
- Flask: Web framework đơn giản, dễ dùng, nhưng chậm
- Tornado
- CherryPy
- Sanic: Nhanh, cộng đồng phát triển nhanh, chưa hỗ trợ Windows

# Cài đặt Sanic

---

# Cài đặt



1. Chuẩn bị hệ điều hành Linux hoặc MacOSX. Nếu đang dùng Windows hãy cài Ubuntu ảo hoá
2. Cài đặt Python 3.7 bản mới nhất có thể
3. Tạo Python Project bằng PyCharm hoặc tạo VirtualEnv
4. cd vào thư mục dự án, kích hoạt VirtualEnv
5. `pip install sanic`

```
from sanic import Sanic
from sanic import response
```

```
app = Sanic()
```

```
@app.route("/")
async def homepage(request):
    return response.html("Hello World")
```

```
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8000, debug=True)
```


Hứng ở cổng 8000

debug and hot reload

```
from sanic import Sanic
from sanic import response
```

```
app = Sanic()
```

```
@app.route("/")
```



Đường dẫn tương đối, relative path

```
async def test(request):
```

```
    return response.json(
```

```
        [{
```

```
            "first_name": "Geno",
```

```
            "last_name": "Leitch",
```

```
            "email": "gleitche0@newsvine.com"
```

```
        ], {
```

```
            "first_name": "Onofredo",
```

```
            "last_name": "Newtown",
```

```
            "email": "onewtown1@homestead.com"
```

```
        ]])
```

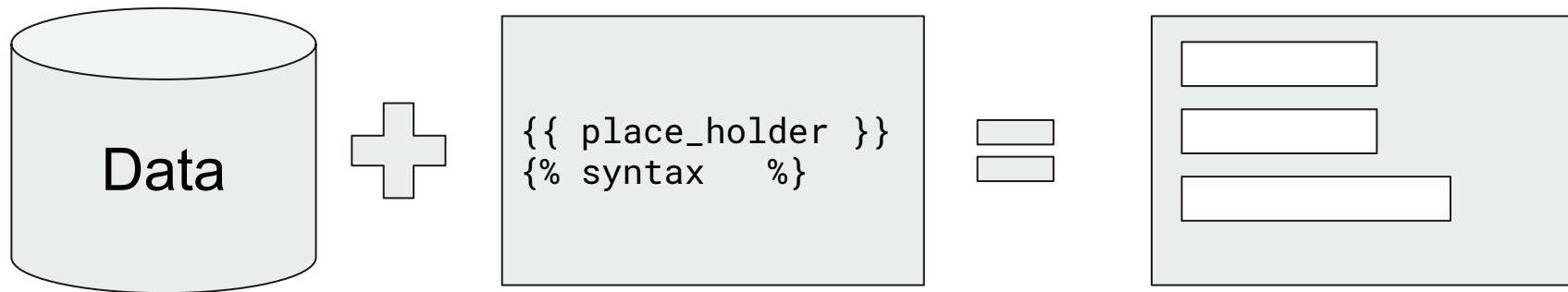
Trả về JSON

```
if __name__ == "__main__":
```

```
    app.run(host="0.0.0.0", port=8000)
```



# View Template



View Template là những khuôn mẫu để đổ dữ liệu vào tạo ra giao diện hoàn chỉnh trả về

View Template loại bỏ việc hard code giao diện với dữ liệu → Decoupling

```
from sanic import Sanic
from sanic_jinja2 import SanicJinja2  # pip install sanic_jinja2
```

```
app = Sanic()
jinja = SanicJinja2(app)
```

Cần phải cài đặt sanic\_jinja2

```
@app.route('/')
@jinja.template('index.html')  # decorator method is static method
async def index(request):
    return {'greetings': 'Hello, sanic!',
            'users': [{ 'url': 'https://techmaster.vn',
                          'username': 'Cuong' },
                      { 'url': 'https://google.com',
                          'username': 'Sergey Bin' }
            ]}
```

```
# Chú ý là file template luôn phải để trong thư mục templates
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000, debug=True)
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Hello</title>
</head>
<body>
  <h1>{{greetings}}</h1>
  <ul>
    {% for user in users %}
      <li><a href="{{ user.url }}">{{ user.username }}</a></li>
    {% endfor %}
  </ul>
</body>
</html>
```

place holder

Loop

---

# Một vài xử lý căn bản

# Chuyển hướng request

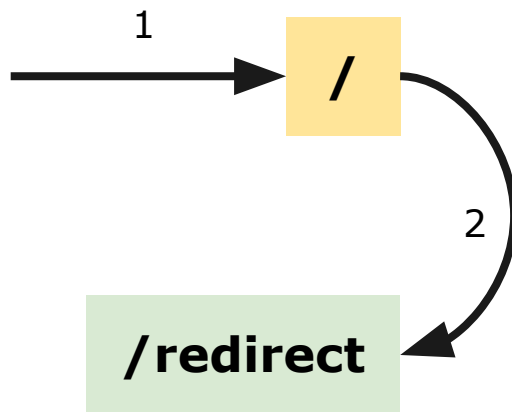
```
from sanic import Sanic
from sanic import response
```

```
app = Sanic(__name__)
```

```
@app.route('/')
def handle_request(request):
    return response.redirect('/redirect')
```

```
@app.route('/redirect')
async def test(request):
    return response.json({"Redirected": True})
```

```
if __name__ == '__main__':
    app.run(host="0.0.0.0", port=8000)
```



# 2 loại web method chính để xử lý request

	GET	POST
Mục đích	Gửi đường dẫn + tham số để lấy dữ liệu	Gửi đường dẫn + dữ liệu lên server
Kiểu dữ liệu	Dữ liệu gửi qua query string trên URL <code>http://hostname/path?querystring</code>	Dữ liệu gói trong header của request
Kích thước dữ liệu	Giới hạn hoặc 8kb	Giới hạn lớn hơn nhiều, thậm chí có thể tải lên file binary
Bảo mật	Không	Có: mã hóa
Nên	Chỉ đọc dữ liệu từ server	Thêm mới, cập nhật. Nếu muốn xóa hãy dùng DELETE

# Hững GET request

  
`https://www.xyz.com/path?key1=value1&key2=value2`

hostname

query string

```
from sanic import Sanic
from sanic_jinja2 import SanicJinja2
from sanic.response import json

app = Sanic()

jinja = SanicJinja2(app)

movies = [
    {"id": 1, "title": "Gone with wind"},
    {"id": 2, "title": "Về nhà đi con"},
    {"id": 3, "title": "Người nhện xa nhà"},
]

@app.route('/')
@jinja.template('index.html')
async def index(request):
    return
```



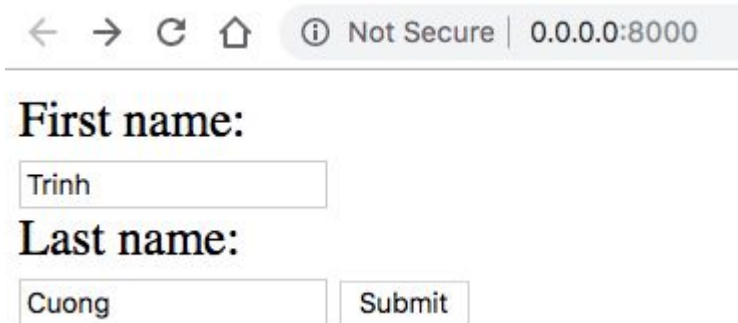
```
@app.route('/movies/all')
async def get_movie(request):
    return json(movies)
```

```
@app.route('/movies')
async def get_movie(request):
    print(request.args["id"][0])
    print(request.query_args)
    return json(request.args)
```

```
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000, debug=True)
```

# Xử lý form

```
<form action="" method="POST">
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
  <input type="submit" value="Submit">
</form>
```



A screenshot of a web browser window. The address bar shows "Not Secure" and "0.0.0.0:8000". The page content includes a form with two text input fields and a submit button. The first field is labeled "First name:" and contains the text "Trinh". The second field is labeled "Last name:" and contains the text "Cuong". A "Submit" button is located to the right of the last name field.

After posted

```
{
  "firstname": [
    "Trinh"
  ],
  "lastname": [
    "Cuong"
  ]
}
```

```
from sanic import Sanic
from sanic_jinja2 import SanicJinja2  # pip install sanic_jinja2
from sanic.response import json

app = Sanic()
jinja = SanicJinja2(app)

@app.route('/')
@jinja.template('index.html')  # decorator method is static method
async def index(request):
    return
```

```
@app.route('/', methods=['POST'])
@jinja.template('index.html')
async def post_index(request):
    return json(request.form)
```

```
# Chú ý là file template luôn phải để trong thư mục templates
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000, debug=True)
```

# REST API

---