

Dasar Pemrogramman Backend

Oleh : Rudi Hartono, M.Kom

Prodi - S1 Teknik Informatika
STIKOM PGRI Banyuwangi

Pengenalan Flask & ReST-API



Pengenalan Framework Python FLASK

❖ Apa Itu Python Flask?

- Python Flask adalah kerangka kerja (*framework*) *web* ringan dan sederhana yang ditulis dalam bahasa pemrograman Python.
- Flask dianggap ringan karena tidak menuntut banyak hal dari sistem atau memerlukan banyak dependensi dibandingkan kerangka kerja Python lainnya. Dengan kata lain, Flask memberikan penggunanya fleksibilitas dan kontrol lebih banyak untuk membangun aplikasi *web* sesuai kebutuhan dan keinginan masing-masing.
- Python Flask menjadi framework yang populer di kalangan *web developer* karena kemudahan penggunaan dan fleksibilitasnya. Hal ini dibuktikan dengan adanya komunitas pengguna Flask yang besar. Komunitas ini selalu siap membantu pengguna baru memecahkan masalah dan memahami cara kerja Flask kapan pun.

❖ Pembuatan Virtual Environments

- Virtual environment adalah sebuah alat untuk menjaga ruang terpisah untuk sebuah proyek dengan pustaka dan dependensi di satu tempat. Environment ini spesifik ke proyek tertentu dan tidak berinterfer dengan dependensi proyek lainnya.
- Untuk membuat Virtual Environments , pertama kita harus masuk ke folder tempat proyek kita buat kemudian ketikkan perintah dibawah ini :

```
python -m venv .venv
```

❖ Install Flask

- Flask merupakan framework yang dikembangkan dalam bahasa pemrograman Python, sehingga Python menjadi prasyarat utama untuk menggunakannya
- Untuk instalasi Flask dapat dilakukan dengan perintah PIP pada folder yang sudah kita buat.

```
pip install flask
```

❖ Pengujian Flask

Untuk menguji apakah framework flask sudah terinstall dengan baik, maka kita coba membuat aplikasi kecil untuk menampilkan kata “SELAMAT DATANG”

- ✓ Buat folder tempat proyek yang akan kita buat
- ✓ Instal Virtual Environment kedalam folder tersebut (lewat terminal vscode)
- ✓ Install flask kedalam folder tersebut (lewat terminal vscode)
- ✓ Buat file python dengan nama “**selamatdatang.py**” kemudian tuliskan script dibawah ini

```
1  from flask import Flask
2  app = Flask(__name__)
3  app.config["DEBUG"] = True
4
5  @app.route('/')
6  def home():
7      return 'Selamat Datang'
8
9  app.run()
10
```

- ✓ Untuk menjalankan file tersebut lewat terminal vscode ketikan perintah dibawah ini

python selamatdatang.py

- ✓ Jalankan link URL yang sudah di buatkan oleh python dengan (ctrl+click)

```
PS F:\Materi Kuliah\Ganjil 2024-2025\python\latihanflas
* Serving Flask app 'selamatdatang'
* Debug mode: off
WARNING: This is a development server. Do not use it in
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
```



Pengenalan ReST - API

❖ Apa Itu API ?

- **API (Application Programming Interface)** adalah Antarmuka pemrograman aplikasi adalah penerjemah komunikasi antara klien dengan server untuk menyederhanakan implementasi dan perbaikan software.
- Bisa diartikan juga sebagai sekumpulan perintah, fungsi, serta protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu.

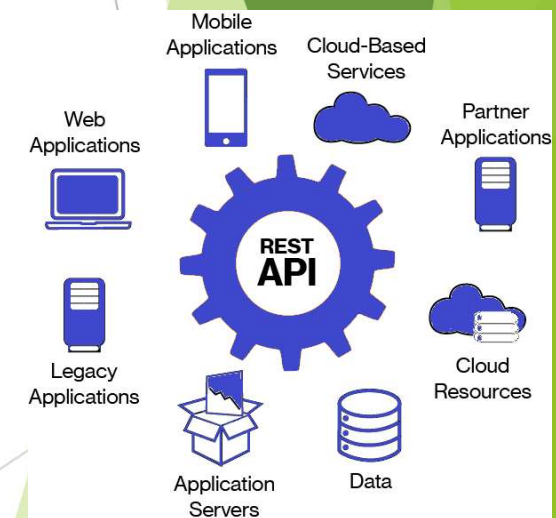
❖ Apa Itu REST ?

REST (REpresentational State Transfer) merupakan gaya arsitektural perangkat lunak yang di dalamnya mendefinisikan aturan-aturan untuk membuat web

REST ini berbentuk **JSON** object data. Yang dapat mempermudah client untuk melakukan parsing data dan tergolong lebih ringan.

❖ Apa Itu REST API ?

- **REST API**, atau **Representational State Transfer Application Programming Interface**, adalah suatu jenis antarmuka yang digunakan untuk mengizinkan komunikasi dan pertukaran data antara berbagai perangkat lunak atau sistem di lingkungan jaringan komputer. REST API didasarkan pada prinsip-prinsip arsitektur REST, yang mengutamakan sederhana, skalabilitas, dan keterbacaan. Cara kerja REST API adalah sebagai berikut:
- **REST API** menggunakan metode **HTTP (GET, POST, PUT, DELETE)** dan URL untuk berinteraksi dengan sumber daya. Permintaan klien mencakup metode HTTP, URL, dan informasi tambahan seperti parameter, header, atau body. Server REST API menyediakan sumber daya yang diidentifikasi melalui URL. Balasan server mencakup status HTTP, data, dan informasi tambahan seperti header setelah memproses permintaan klien.



❖ HTTP Method

➤ **HTTP (*Hypertext Transfer Protocol*)** adalah sebuah jalan atau cara komunikasi dapat terjalin antara client dengan server.

Dalam HTTP Method REST yang sering digunakan antara lain:

- ❑ **GET** : Biasanya fungsi ini digunakan untuk perintah menampilkan data.
- ❑ **POST** : Fungsi ini biasanya digunakan untuk perintah dalam menambahkan data.
- ❑ **PUT** : Fungsi ini biasanya digunakan untuk perintah edit data
- ❑ **DELETE** : Fungsi ini biasanya digunakan untuk perintah dalam menghapus data.



➤ Kita akan coba sederhanakan bahasanya tentang **REST API** , jadi API disini berperan sebagai perantara untuk ngobrol dengan server, jadi untuk akses data di sebuah server dilakukan tidak secara direct melainkan melalui API ini, sedangkan REST ini sebagai aturan-aturan / perintah apa saja yang diperbolehkan untuk dilakukan oleh client melalui protokol http.

❖ Studi Kasus

Pertama kita akan coba membuat sebuah file dengan nama “app.py” kemudian ketikkan perintah kemudian jalankan perintah di terminal `python app.py`

```
app.py X...
1  from flask import Flask, jsonify, request, make_response
2
3  app = Flask(__name__)
4  app.config["DEBUG"] = True
5
6  @app.route('/', methods=['GET'])
7  def hello():
8      return "Hello World"
9
10 app.run()
```

PROBLEMS 25 OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell

Copyright (c) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/powershell>

PS E:\Python> python app.py

- * Serving Flask app "app" (lazy loading)
- * Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
- * Debug mode: on
- * Restarting with stat
- * Debugger is active!
- * Debugger PIN: 248-421-238
- * Running on <http://127.0.0.1:5000/> (Press CTRL+C to quit)

Pada tahapan pertama ini kita sedang menggunakan method GET untuk menampilkan data dari API, Setelahnya kita akan mencoba dengan mengubah dengan bentuk JSON format.

```
app.py > ...
1  from flask import Flask, jsonify, request, make_response
2
3  app = Flask(__name__)
4  app.config["DEBUG"] = True
5
6  @app.route('/', methods=['GET'])
7  def hello():
8      data = [{
9          'nama': 'Galih',
10         'pekerjaan': 'Web Engineer',
11         'pesan': 'Hello World'
12     }]
13     return make_response(jsonify({'data': data}), 200)
14
15  app.run()
```

Pada pengertian `return make_response(jsonify({'data': data}), 200)` memiliki arti melakukan pengembalian data / response dengan format json dengan status code 200 (yang berarti data suksse / oke). disini kita dapat mengganti dengan status code lainnya seperti 400, 500, 404 dll [lihat detail], ini merupakan sebuah kode yang dapat dipahami oleh sebuah sistem pada http protokol.

```
16 @app.route('/karyawan', methods=['GET', 'POST', 'PUT', 'DELETE'])
17 def karyawan():
18     try:
19         if request.method == 'GET':
20             data = [{
21                 'nama': 'Galih GET',
22                 'pekerjaan': 'Web Engineer',
23                 'usia': '27',
24             }]
25         elif request.method == 'POST':
26             data = [{
27                 'nama': 'Galih POST',
28                 'pekerjaan': 'Web Engineer',
29                 'usia': '27',
30             }]
31         elif request.method == 'PUT':
32             data = [{
33                 'nama': 'Galih PUT',
34                 'pekerjaan': 'Web Engineer',
35                 'usia': '27',
36             }]
37         else:
38             data = [{
39                 'nama': 'Galih DELETE',
40                 'pekerjaan': 'Web Engineer',
41                 'usia': '27',
42             }]
43     except Exception as e:
44         return make_response(jsonify({'error': str(e)}), 400)
45     return make_response(jsonify({'data': data}), 200)
46
47 app.run()
```

Routes merupakan sebuah endpoint yang akan kita panggil melalui client service. bilamana kita melihat dikodingan pertama kita menulis seperti ini.

```
@app.route('/', methods=['GET'])
```

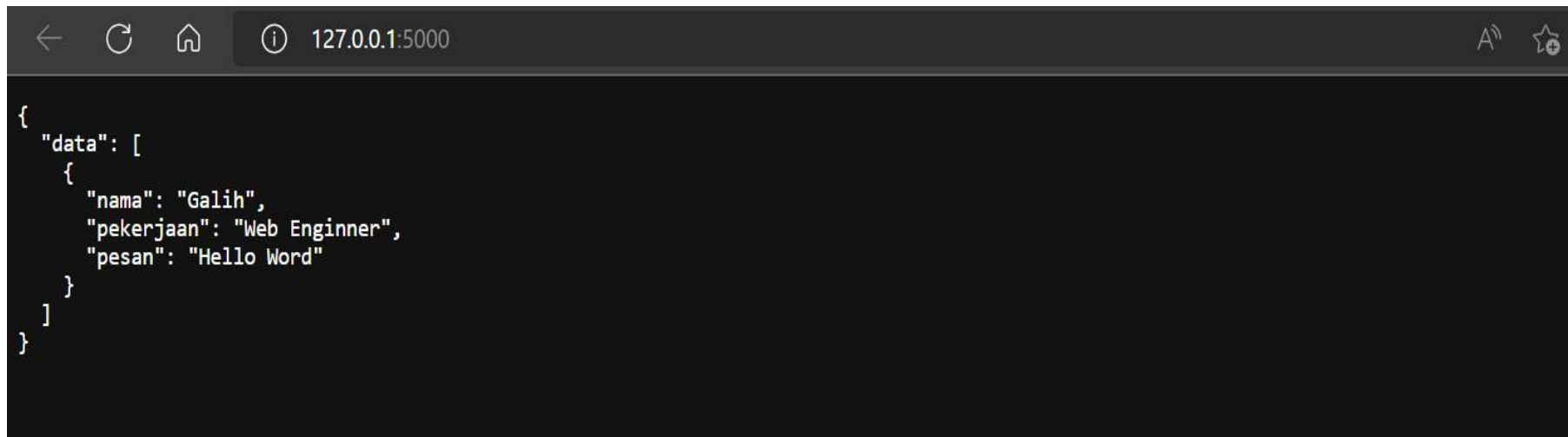
Artinya kita memanggil root direktori atau alamat web pada root domain dan hanya mengizinkan method GET, jadi bila mana pada endpoint ini ada method selain GET maka akan di tolak. Contoh lain:

```
@app.route('/karyawan', methods=['GET', 'POST', 'PUT', 'DELETE'])
```

Artinya endpoint domain/karyawan hanya menerima method GET, POST, PUT dan DELETE, selain itu maka akan ditolak.

if request.method=='GET'

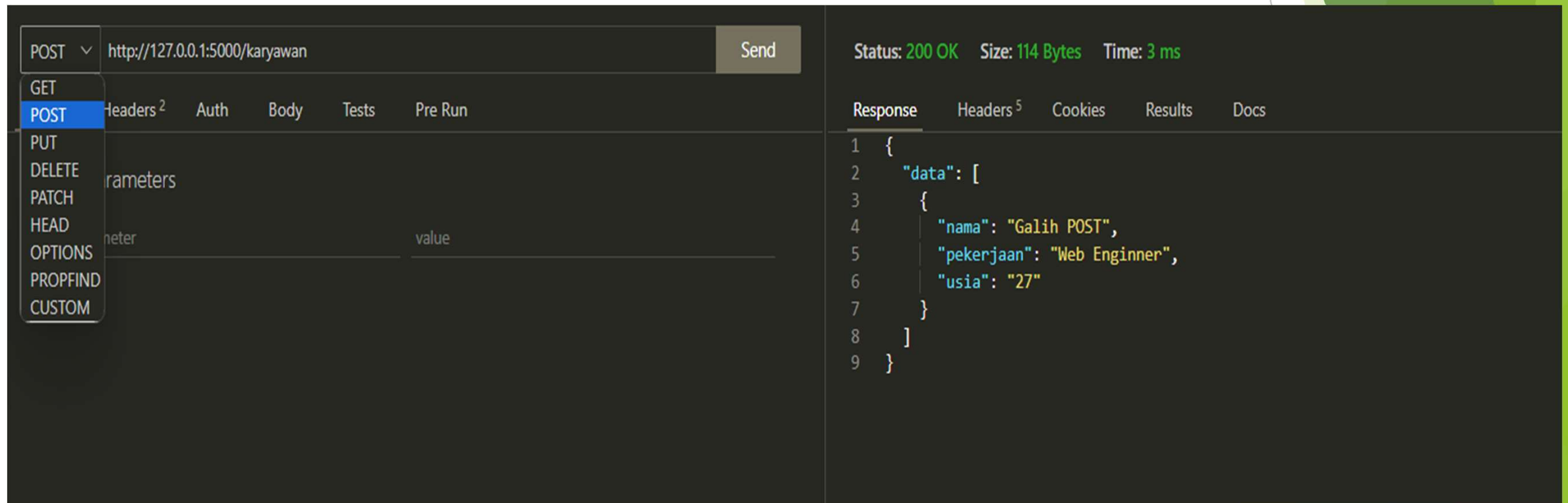
Pada kodingan diatas menjelaskan jika ada request yang tipe methodnya GET maka tampilkan data. sehingga bilamana kita jalankan program diatas maka akan mendapatkan hasil yang berbeda-beda setiap methodnya.

A screenshot of a web browser window with a dark theme. The address bar shows the URL '127.0.0.1:5000'. The main content area displays a JSON object: { "data": [{ "nama": "Galih", "pekerjaan": "Web Enginner", "pesan": "Hello Word" }] }. The browser interface includes back, forward, and home buttons on the left, and volume and star icons on the right.

```
{
  "data": [
    {
      "nama": "Galih",
      "pekerjaan": "Web Enginner",
      "pesan": "Hello Word"
    }
  ]
}
```


Pengujian ReST-API Dengan Thunder Client VS Code

- **Thunder Client** adalah Visual Studio Code Extension yang fiturnya sama seperti Postman, tapi jauh lebih ringan dan practical, terlebih untuk developer yang menggunakan editor Visual Studio Code.
- Untuk menginstall extension Thunder Client dapat dilakukan dengan menekan “Ctrl + Shift + x” , kemudian ketikan Thunder Client dan lakukan penginstalan
- Untuk menjalankan dapat dilakukan dengan menekan icon Thunder client serta ketik “New Request”





Selamat Berkarya & Semoga Sukses