

Nama : Muhammad Irsyad

NIM : 2211102048

Kelas : S1-IF-10-Q

	Institut Teknologi Telkom Purwokerto FAKULTAS INFORMATIKA PROGRAM STUDI S1 TEKNIK INFORMATIKA				
	RENCANA PENUGASAN TERSTRUKTUR MAHASISWA				
MATA KULIAH	Pemrograman Fungsional				
KODE	IF123231	SKS	3	Semester	5
DOSEN PENGAMPU	Andi Prademon Yunus, ST., M.Sc. Eng., Ph.D				
BENTUK TUGAS	Tugas Terstruktur				
WAKTU Pengerjaan Tugas	2 Minggu				
JUDUL TUGAS					
Tugas Terstruktur : Membuat Restfull API dengan tema Bebas					
SUB CAPAIAN PEMBELAJARAN MATA KULIAH					
Mahasiswa mampu menerapkan Fungsional Programming dalam Web Services (C4)					
DESKRIPSI TUGAS					
Mahasiswa diminta untuk membuat RESTful API dengan tema bebas tidak boleh tentang mahasiswa seperti contoh yang sudah diberikan. Skema Method RESTful API yaitu ada GET, POST, PUT, DELETE					
METODE Pengerjaan Tugas					
Berikut adalah panduan poin yang harus disertakan dalam video: <ol style="list-style-type: none">1. Kerjakan build RESTful API menggunakan Flask Framework method GET, POST, PUT, DELETE dengan tema bebas.2. Database yang digunakan bebas.3. Harus sudah terkoneksi dengan database local atau sever4. Berhasil deploy ke server dapat nilai bonus (tidak wajib)					
BENTUK DAN FORMAT LUARAN					

A. Bentuk Pembuatan RESTful API bersifat individu	
B. Format Pengumpulan <ol style="list-style-type: none"> 1. Dokumentasi API Postman 2. Link publish documentation API Postman 	
INDIKATOR, KRITERIA DAN BOBOT PENILAIAN	
Implementasi RESTful API	30
Connect ke database	20
Documentation API	30
Keberagaman Tema yang diambil	20
Total	100
Deploy server (tidak wajib)	+20
DATELINE PENGUMPULAN TUGAS	
Pengumpulan tugas besar Jumat 24 Januari 2024, jam 23: <div>9</div>	
SANKSI DAN PELANGGARAN	
- Terlambat mengumpulkan tidak akan dinilai	
DAFTAR RUJUKAN	

Jawab :

Disini saya Membuat RESTful API untuk menyimpan data Pegawai Negeri sipil

- Database yang digunakan : Localhost

```
1  from flask import Flask, render_template, request, url_for, flash
2  from werkzeug.utils import redirect
3  from flask_mysql import MySQL
4
5  app = Flask(__name__)
6  app.secret_key = 'many random bytes'
7
8  app.config['MYSQL_HOST'] = 'localhost'
9  app.config['MYSQL_USER'] = 'root'
10 app.config['MYSQL_PASSWORD'] = ''
11 app.config['MYSQL_DB'] = 'db_pegawai'
12
13 mysql = MySQL(app)
14
15 @app.route('/')
16 def Index():
17     cur = mysql.connection.cursor()
18     cur.execute("SELECT * FROM pegawai")
19     data = cur.fetchall()
20     cur.close()
21
22     return render_template('index.html', pegawai=data)
23
24 @app.route('/insert', methods = ['POST'])
25 def insert():
```

1. Import:

- from flask import Flask, render_template, request, url_for, flash: Mengimport fungsi-fungsi dari Flask untuk membuat aplikasi web, menampilkan template, menangani permintaan, membuat URL, dan menampilkan pesan flash.
- from werkzeug.utils import redirect: Mengimport fungsi redirect untuk mengarahkan pengguna ke halaman lain.
- from flask_mysql import MySQL: Mengimport ekstensi untuk menghubungkan Flask dengan database MySQL.

2. Membuat Aplikasi Flask:

- app = Flask(__name__): Membuat objek aplikasi Flask.

- `app.secret_key = 'many random bytes'`: Mengatur kunci rahasia untuk mengamankan pesan flash dan fitur-fitur lainnya.

3. Konfigurasi Database MySQL:

- `app.config['MYSQL_HOST'] = 'localhost'`: Mengatur host database.
- `app.config['MYSQL_USER'] = 'root'`: Mengatur nama pengguna database.
- `app.config['MYSQL_PASSWORD'] = ''`: Mengatur kata sandi database (kosong dalam contoh ini).
- `app.config['MYSQL_DB'] = 'db_pegawai'`: Mengatur nama database.mysql
- `MySQL(app)`: Membuat objek MySQL untuk terhubung ke database.

```

14
15 @app.route('/')
16 def Index():
17     cur = mysql.connection.cursor()
18     cur.execute("SELECT * FROM pegawai")
19     data = cur.fetchall()
20     cur.close()
21
22     return render_template('index.html', pegawai=data)
23
24 @app.route('/insert', methods = ['POST'])
25 def insert():
26     if request.method == "POST":
27         flash("Data Inserted Successfully")
28         nama = request.form['nama']
29         nip = request.form['nip']
30         tgl_lahir = request.form['tgl_lahir']
31         pangkat = request.form['pangkat']
32         golongan = request.form['golongan']
33         jabatan = request.form['jabatan']
34         cur = mysql.connection.cursor()
35         cur.execute("INSERT INTO pegawai (nama, nip, tgl_lahir, pangkat, golongan, jabatan) VALUES (%s, %s, %s, %s, %s, %s)",
36                     (nama, nip, tgl_lahir, pangkat, golongan, jabatan))
37         mysql.connection.commit()
38         return redirect(url_for('Index'))
39
40 @app.route('/delete<string:id_data>', methods = ['DELETE'])
41 def delete(id_data):
42     flash("Record Has Been Deleted Successfully")
43     cur = mysql.connection.cursor()
44     cur.execute("DELETE FROM pegawai WHERE id=%s", (id_data,))
45     mysql.connection.commit()
46     return redirect(url_for('Index'))
47
48 @app.route('/update', methods= ['PUT', 'GET'])
49 def update():
50     if request.method == 'PUT':
51         id_data = request.form['id']
52         nama = request.form['nama']
53         nip = request.form['nip']
54         tgl_lahir = request.form['tgl_lahir']
55         pangkat = request.form['pangkat']
56         golongan = request.form['golongan']
57         jabatan = request.form['jabatan']
58
59         cur = mysql.connection.cursor()
60         cur.execute("""
61             UPDATE pegawai SET nama=%s, nip=%s, tgl_lahir=%s, pangkat=%s, golongan=%s, jabatan=%s
62             WHERE id=%s
63             """, (nama, nip, tgl_lahir, pangkat, golongan, jabatan, id_data))
64         mysql.connection.commit()
65         flash("Data Updated Successfully")
66         return redirect(url_for('Index'))
67
68 if __name__ == "__main__":
69     app.run(debug=True)

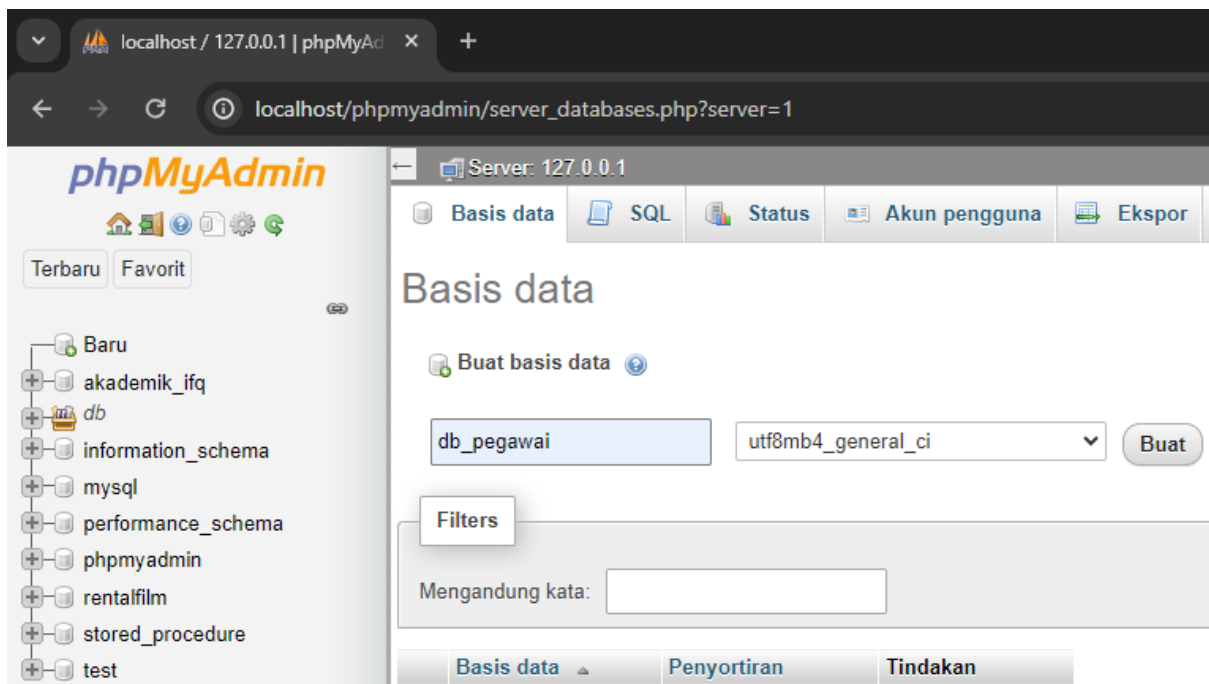
```

4. Fungsi-Fungsi Utama:

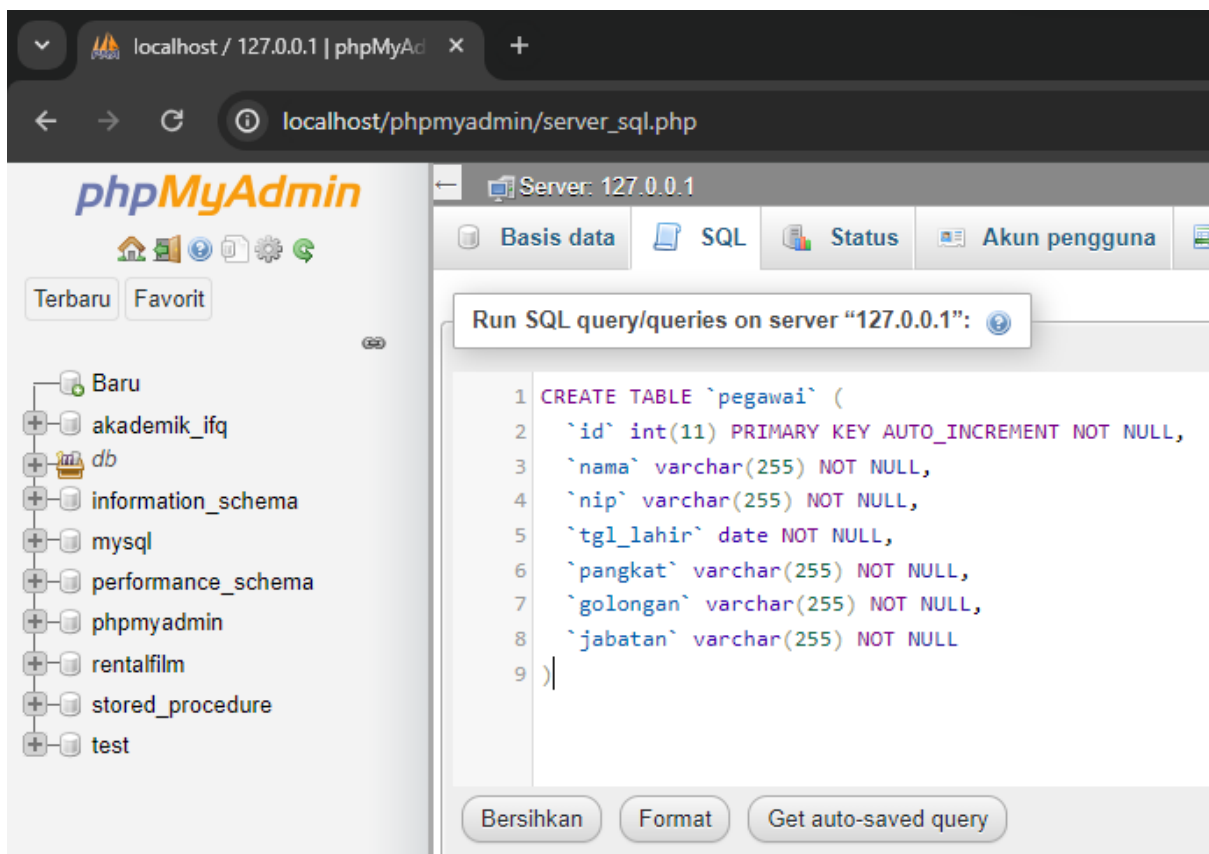
- `Index()`: Menampilkan halaman utama yang menampilkan daftar pegawai.
 - Mengambil data pegawai dari database menggunakan cursor MySQL.
 - Mengirimkan data pegawai ke template `index.html` untuk ditampilkan.
- `insert()`: Menambahkan data pegawai baru ke database.
 - Menerima data pegawai dari form HTML melalui request POST.
 - Memasukkan data pegawai ke dalam database menggunakan SQL INSERT.
 - Menampilkan pesan flash "Data Inserted Successfully".
 - Mengarahkan pengguna kembali ke halaman utama.
- `delete(id_data)`: Menghapus data pegawai dengan ID yang diberikan.
 - Menghapus data pegawai dari database menggunakan SQL DELETE.
 - Menampilkan pesan flash "Record Has Been Deleted Successfully".
 - Mengarahkan pengguna kembali ke halaman utama.
- `update()`: Memperbarui data pegawai yang sudah ada.
 - Menerima data pegawai yang diperbarui dari form HTML melalui request POST.
 - Memperbarui data pegawai di database menggunakan SQL UPDATE.
 - Menampilkan pesan flash "Data Updated Successfully".
 - Mengarahkan pengguna kembali ke halaman utama.

5. Menjalankan Aplikasi:

- `if __name__ == "__main__":`: Memeriksa apakah kode ini dijalankan sebagai program utama.
- `app.run(debug=True)`: Menjalankan aplikasi Flask dalam mode debug, yang memudahkan untuk melihat error dan perubahan.



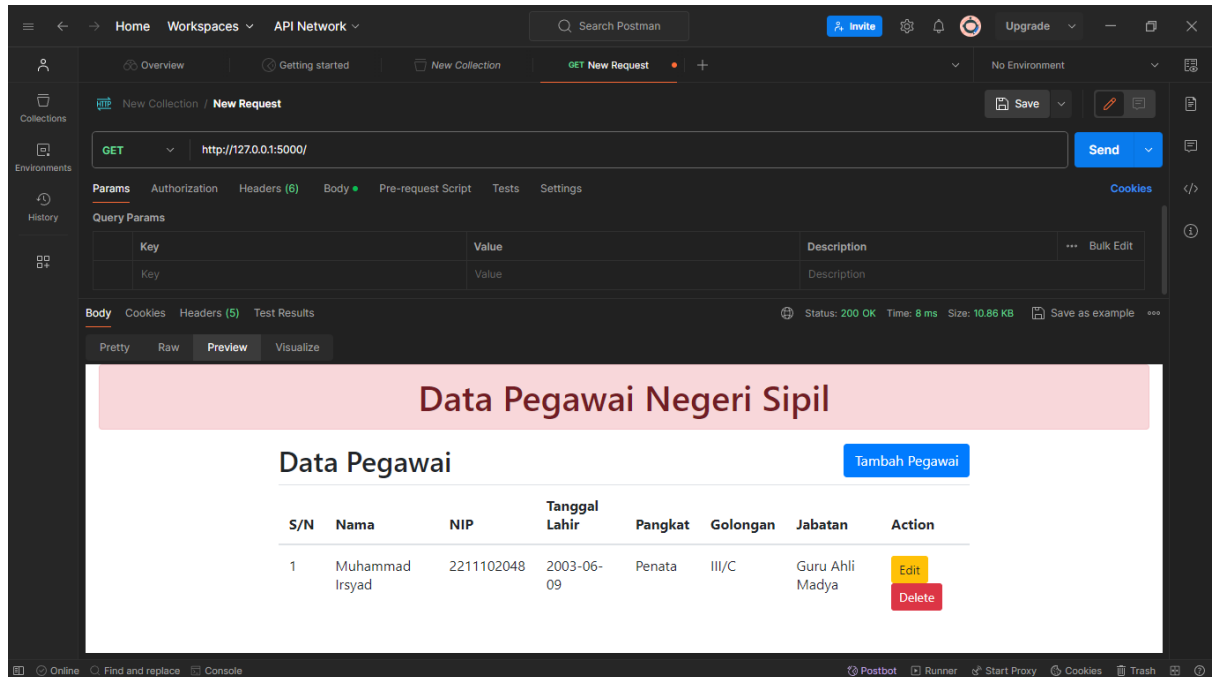
Buat database di phpMyAdmin db_pegawai sesuai dengan kodingan koneksi



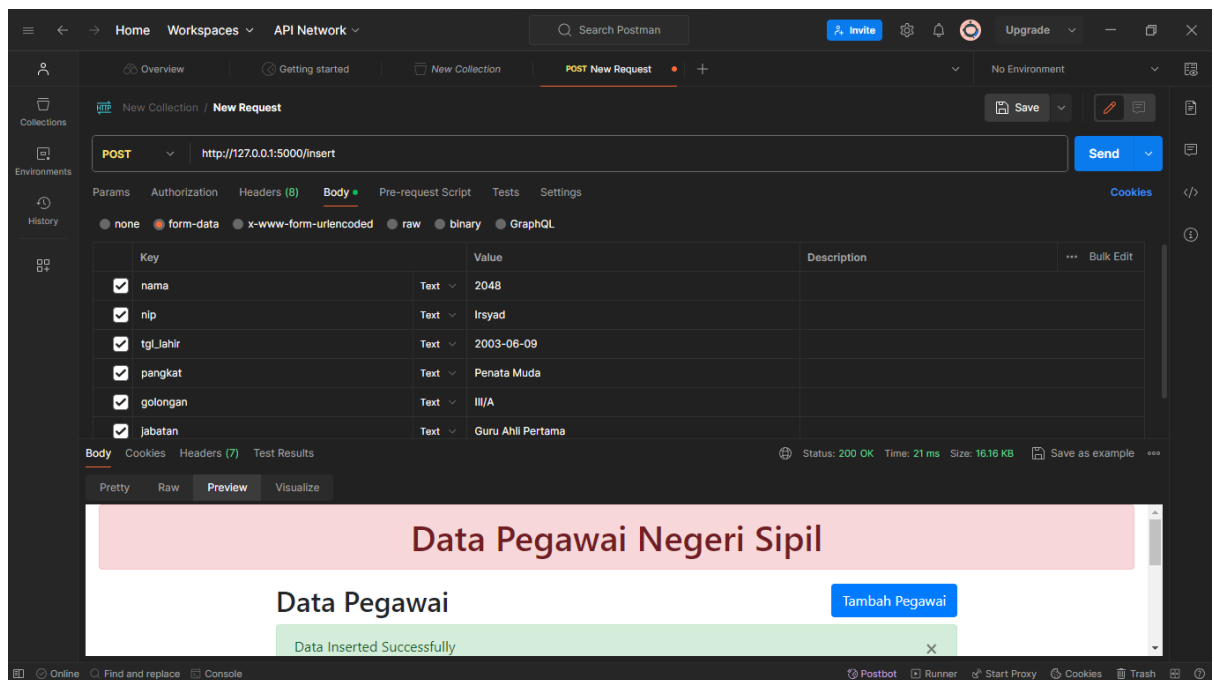
Lalu membuat tabel pegawai dengan kolom id, nama, nip, tgl_lahir, pangkat, golongan, dan jabatan

PENERAPAN POSTMAN

- GET



- POST



• PUT

The screenshot shows the Postman interface for a PUT request to `http://127.0.0.1:5000/update`. The request body is a JSON object with employee data. The response status is 200 OK, and the body is a simple HTML page.

Request Details:

- Method: PUT
- URL: `http://127.0.0.1:5000/update`
- Body (form-data):

Key	Type	Value
Id	Text	2
nama	Text	Irsyad
nip	Text	2048
tgl_lahir	Text	2001-04-02
pangkat	Text	Penata Muda Tingkat 1
golongan	Text	III/C
Jabatan	Text	Guru Ahli Muda

Response: Status: 200 OK, Time: 30 ms, Size: 16.18 KB. The response body is a simple HTML page with the title "Data Pegawai Negeri Sipil" and a button "Tambah Pegawai".

The screenshot shows the Postman interface for a PUT request to `http://127.0.0.1:5000/update`. The request body is a JSON object with employee data. The response status is 200 OK, and the body is a detailed HTML page showing a table of employees.

Request Details:

- Method: PUT
- URL: `http://127.0.0.1:5000/update`
- Body (form-data):

Key	Type	Value
Id	Text	2
nama	Text	Irsyad
nip	Text	2048
tgl_lahir	Text	2001-04-02
pangkat	Text	Penata Muda Tingkat 1
golongan	Text	III/C
Jabatan	Text	Guru Ahli Muda

Response: Status: 200 OK, Time: 30 ms, Size: 16.18 KB. The response body is a detailed HTML page with the title "Data Pegawai Negeri Sipil". It includes a table of employees and a "Tambah Pegawai" button.

S/N	Nama	NIP	Tanggal Lahir	Pangkat	Golongan	Jabatan	Action
1	Muhammad Irsyad	2211102048	2003-06-09	Penata	III/C	Guru Ahli Madya	Edit Delete
2	Irsyad	2048	2001-04-02	Penata Muda Tingkat 1	III/C	Guru Ahli Muda	Edit Delete

• DELETE

The screenshot shows a Postman interface with a DELETE request to `http://127.0.0.1:5000/delete/2`. The response is a 200 OK status with a 34 ms time and 11.35 KB size. The body of the response is displayed in the 'Preview' tab, showing a confirmation message and a table of employee data.

Data Pegawai Negeri Sipil

Data Pegawai [Tambah Pegawai](#)

Record Has Been Deleted Successfully

S/N	Nama	NIP	Tanggal Lahir	Pangkat	Golongan	Jabatan	Action
1	Muhammad Irsyad	2211102048	2003-06-09	Penata	III/C	Guru Ahli Madya	Edit Delete