MODUL 03

POST METHOD

A. Capaian Pembelajaran

Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu:

1. Menangani permintaan HTTP POST pada sisi server menggunakan PHP.
2. Menerapkan penggunaan prepared statement untuk mencegah SQL Injection.
3. Mengembalikan respons HTTP yang sesuai berdasarkan hasil eksekusi operasi database.
4. Menghasilkan output dalam format JSON untuk komunikasi API.

B. Review Materi POST Method

Salah satu prinsip CRUD yang digunakan RESTful API adalah fungsi *Create* atau biasa dinamakan metode POST. POST digunakan ketika ingin membuat atau menambah data baru ke dalam sistem. POST pada RESTful API akan membuat data baru yang dimasukkan oleh pengguna dan mengirimkannya ke server untuk disimpan.

C. Deskripsi Singkat

Modul ini membimbing mahasiswa untuk membuat endpoint POST yang menyimpan data pengguna ke dalam database users menggunakan PHP. Praktikum ini juga membahas pengolahan input, pengamanan melalui prepared statement, dan pengiriman respons dalam format JSON.

D. Persiapan

1. Pastikan telah memiliki web server lokal (XAMPP, Laragon, dsb).
2. Database MySQL sudah aktif.
3. Database **test_db** dan tabel **users** telah dibuat.
4. Telah membuat folder praktikum-api dengan file **db.php** yang berisi koneksi ke database

E. Membuat API POST Method

1. Buat file baru untuk membuat API dengan nama **post.php** di dalam folder **praktikum-api** yang telah dibuat sebelumnya.
2. Set Header response

   Tambahkan instruksi PHP untuk memastikan response dari server berupa JSON:

   ```php
   header('Content-Type: application/json');
   ```

3. Panggil file koneksi database menggunakan **`require`** dan diikuti **nama file koneksi**
4. Deklarasi variabel untuk menyimpan data yang dikirim oleh client menggunakan **$_POST**

   ```php
   $name = $_POST['name'] ?? '';
   ```

5. Buat deklarasi untuk variabel **`$email`** dengan $_POST seperti variabel **`$name`** di atas
6. Buat validasi input untuk variabel name dan email

```php
if ($name && $email) {
    // True
} else {
    // False
}
```

7. Buat prepared statement berupa query mysql untuk menambah data ke dalam database. Buat di dalam validasi input yang dibuat sebelumnya.

```php
$stmt = $conn->prepare("INSERT INTO users (name, email) VALUES (?, ?)");
```

8. Selanjutnya buatlah syntax untuk mengikat tipe data dan variabel yang telah dibuat menjadi sebuah parameter.

```php
$stmt->bind_param("ss", $name, $email);
```

9. Buat syntax untuk mengeksekusi parameter yang telah dibuat dengan menulis **`$stmt`** lalu diikuti dengan **`execute();`**
10. Masih didalam fungsi validasi input, buat pengecekan input data dengan if untuk mengembalikan respons sukses atau gagal.

```php
if ($stmt->affected_rows > 0) {
    // response sukses
} else {
    // response gagal
}
```

11. Pada bagian untuk respons sukses tulislah **`http_response_code(201);`** untuk memberikan kode **201 Created**.
12. Lalu buat kode untuk mengembalikan pesan sukses seperti di bawah ini

```php
echo json_encode([
    // pesan sukses
]);
```

13. Isi pesan sukses dengan
    - **status = success**
    - **message = User berhasil ditambahkan**
    - lalu array data seperti ini

```
'data' => [
    'id' => $stmt->insert_id,
    'name' => $name,
    'email' => $email
]
```

14. Setelah itu tulislah `http_response_code(500);` pada bagian respons gagal. Lalu buat kode untuk mengembalikan pesan gagal seperti di bawah ini

```
echo json_encode([
    // pesan error
]);
```

15. Isi pesan gagal dengan
    - **status = error**
    - **message = Gagal menambahkan user**
16. Tulis kode di bawah ini untuk menutup statement.

```
$stmt->close();
```

17. Setelah menuliskan keseluruhan respons sukses, maka selanjutnya adalah menulis respons gagal untuk mengembalikan pengecekan yang bernilai false (Lihat langkah nomor 6).
18. Pada bagian untuk respons gagal tulislah `http_response_code(400);` untuk memberikan kode **400 Bad Request**.
19. Lalu buat kode untuk mengembalikan pesan gagal seperti di bawah ini

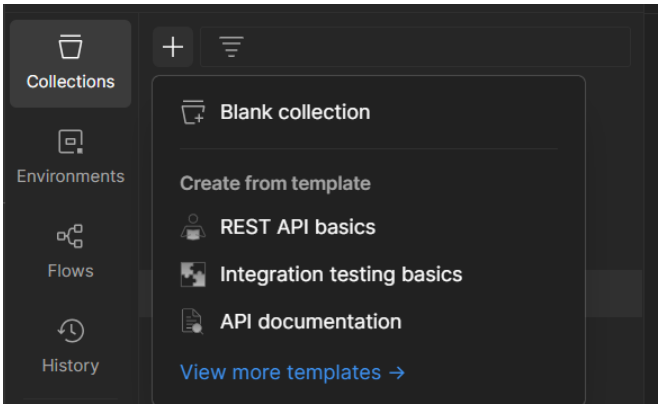
```
echo json_encode([
    // pesan error
]);
```

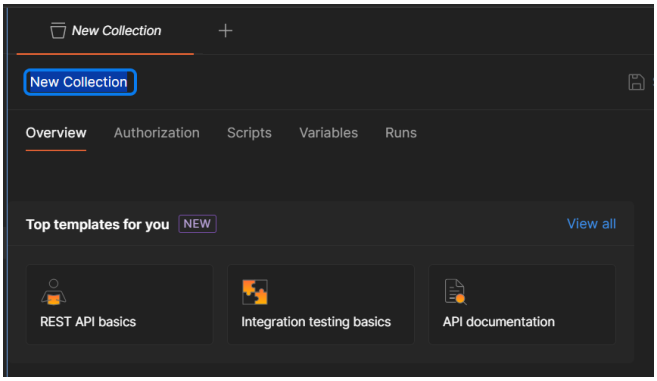
20. Isi pesan error dengan
    - **status = error**
    - **message = Data tidak lengkap**
21. Terakhir tulis kode untuk menutup koneksi dengan database

```
$conn->close();
```

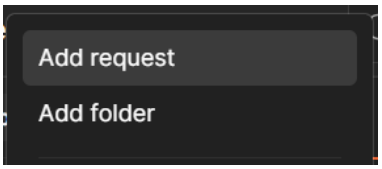
F. Pengujian Postman

1. Buka aplikasi postman yang telah diinstal sebelumnya.
2. Buat **Collection** baru lalu ubah namanya menjadi Praktikum API

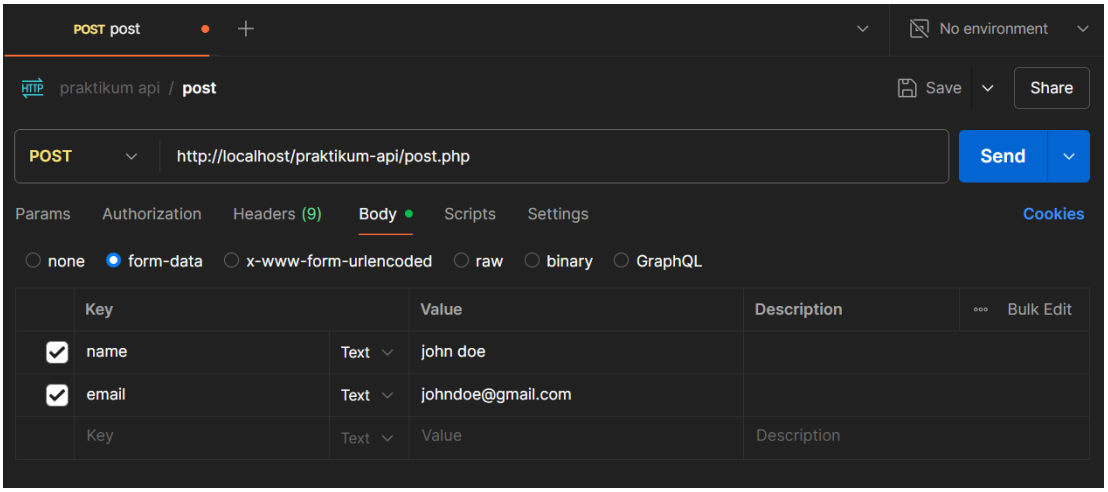




3. Setelah itu klik icon titik tiga pada collection dan pilih **Add Request**



4. Beri nama **post** untuk request yang baru saja dibuat.
5. Selanjutnya atur **method** menjadi **POST** dan ketikkan **url** lokal diikuti lokasi file **post.php**
6. Pilih **Body → form-data**, lalu isi bagian **Key** dengan **nama variabel** yang sesuai dengan API yang dibuat. Isi bagian **Value** dengan data yang ingin dimasukkan ke database



7. Setelah mengisi **Value**, klik tombol **Send** di pojok kanan atas. Respons dari API akan muncul seperti gambar berikut