MODUL 05

RESTful API - PUT

Materi

1. RESTful API

REST *(Representational State Transfer)* adalah arsitektur API *(Application Programming Interface)* yang menyediakan komunikasi *client-server* untuk aplikasi web melalui protokol HTTP. RESTful API merupakan implementasi dari REST API yang benar-benar mengikuti semua prinsip REST dengan baik. RESTful API beroperasi berdasarkan prinsip CRUD *(Create, Read, Update, Delete)*.

2. PUT

Salah satu prinsip CRUD yang digunakan RESTful API adalah fungsi *PUT* atau biasa dinamakan metode PUT. PUT digunakan ketika ingin mengubah nilai data yang ada dalam sistem. PUT pada RESTful API akan mengambil data yang ditentukan berdasarkan id untuk dilakukan perubahan data dan mengirimkannya kembali ke server untuk disimpan.

Deskripsi Tugas

Pada praktikum ini, mahasiswa akan mempraktikkan pembuatan fungsi PUT dalam RESTful API. Mahasiswa akan belajar memahami struktur kode program yang paling mendasar dari pembuatan RESTful API dengan fungsi PUT.

Praktikum

No.	Langkah-langkah
1.	Pertama buat buat file dengan nama “post.php” di dalam folder “api” yang telah dibuat sebelumnya `<?php` `header("Content-Type: application/json");` `require "data.php";` **`header("Content-Type: application/json");`** ➢ Menetapkan **header response** agar output yang dikirimkan ke client berupa **JSON**.

	`require "data.php";` ➢ Mengimpor file **data.php**, yang berisi fungsi **`getData()`** dan **`saveData()`**, digunakan untuk membaca dan menyimpan data dalam format JSON.
2.	```php $data = getData(); $id = isset($_GET['id']) ? $_GET['id'] : null; $input = json_decode(file_get_contents("php://input"), true); ``` **`$data = getData();`** ➢ Memanggil fungsi **`getData()`** untuk membaca isi file JSON dan menyimpannya dalam variabel **`$data`** sebagai array. **`$id = isset($_GET['id']) ? $_GET['id'] : null;`** ➢ Mengecek apakah ada parameter id yang dikirim melalui **query string** (misalnya ?id=1). Jika id ada, nilainya disimpan ke dalam **`$id`**, jika tidak, maka **`$id`** akan bernilai null. **`json_decode(..., true)`** ➢ Mengubahnya menjadi **array PHP**.
3.	```php if (!$id \|\| !isset($data[$id])) { http_response_code(404); echo json_encode(["status" => 404, "error" => "Item Not Found"]); exit; } ``` Jika **tidak ada id** dalam query string ATAU **id tidak ditemukan** dalam data JSON, API akan mengembalikan error **404 Not Found**.
4.	```php if (!isset($input['name'])) { http_response_code(400); echo json_encode(["status" => 400, "error" => "Invalid Input"]); exit; } ``` Jika body request tidak berisi **name**, API mengembalikan **400 Bad Request**.
5.	```php $data[$id]['name'] = $input['name']; saveData($data); ```

	`$data[$id]['name'] = $input['name];` ➢ Mengupdate nama produk berdasarkan id yang diberikan. **`saveData($data);`** ➢ Menyimpan data baru ke JSON.
6.	 Jika update berhasil, API mengembalikan **200 OK** dengan detail produk yang telah diperbarui.

```php
http_response_code(200);
echo json_encode([
    "status" => 200,
    "message" => "Item Updated",
    "produk" => $data[$id]
]);
```