	MODUL PRAKTIKUM	JTI
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	Hal 1 of 12

TUJUAN PRAKTIKUM

1. Mahasiswa mampu memahami konsep RESTful API
2. Mahasiswa mampu menerapkan RESTful API di framework
3. Mahasiswa mampu menguji RESTful API di framework

TEORI SINGKAT

REST API adalah salah satu teknologi API yang sering kita gunakan ketika akan mengembangkan sebuah Web APIs.

REST API sendiri bersifat stateless. Dimana setiap kali request harus menyertakan semua data dan parameter dengan lengkap ketika mengakses suatu endpoint.

Pada arsitektur REST, REST server menyediakan resources (sumber daya/data) dan REST client mengakses dan menampilkan resource tersebut untuk penggunaan selanjutnya. Setiap resource diidentifikasi oleh URIs (Universal Resource Identifiers) atau global ID. Resource tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML.

Web service yang berbasis arsitektur REST kemudian dikenal sebagai RESTful web services. Layanan web ini menggunakan metode HTTP untuk menerapkan konsep arsitektur REST.

Sebuah client mengirimkan sebuah data atau request melalui HTTP Request dan kemudian server merespon melalui HTTP Response.

REST memiliki standarisasi dalam pemakaian, yaitu URL dan HTTP method. HTTP method digunakan untuk mengetahui kegunaan dari URL yang diakses, sehingga mempermudah dalam penulisan URL.

Berikut adalah jenis HTTP method yang sering digunakan :

1. GET, digunakan untuk membaca sebuah record atau daftar record
2. POST, digunakan untuk menambah sebuah record
3. PUT, digunakan untuk mengubah semua field dalam sebuah record
4. PATCH, digunakan untuk mengubah beberapa field dalam sebuah record
5. DELETE, digunakan untuk menghapus sebuah record.

	MODUL PRAKTIKUM	JTI
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	Hal 2 of 12

Komponen dari HTTP Request :

- Verb, HTTP method yang digunakan misalnya GET, POST, DELETE, PUT dll.
- Uniform Resource Identifier (URI) untuk mengidentifikasi lokasi resource pada server.
- HTTP Version, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
- Request Header, berisi metadata untuk HTTP Request. Contoh, type client/browser, format yang didukung oleh client, format dari body pesan, seting cache dll.
- Request Body, konten dari data.

Komponen dari HTTP Response :

- Status/Response Code, mengindikasikan status server terhadap resource yang direquest. misal : 404, artinya resource tidak ditemukan dan 200 response OK.
- HTTP Version, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
- Response Header, berisi metadata untuk HTTP Response. Contoh, type server, panjang content, tipe content, waktu response, dll
- Response Body, konten dari data yang diberikan.

PRAKTIKUM

Buatlah projek laravel baru dan beri nama **mahasiswa_api** dengan cara menjalankan script berikut pada cmd atau comand prompt yang sudah terarah ke folder htdocs yang ada pada xampp.

```
cd ../../xampp/htdocs
```

```
composer create-project --prefer-dist laravel/laravel mahasiswa_api
```

	MODUL PRAKTIKUM	JTI Hal 3 of 12
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	

KONEKSI DATABASE MYSQL

Konfigurasi database pada file .env yang ada pada proyek Laravel mahasiswa_api/.env menjadi seperti pada gambar dibawah:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=db_crudlaravel
DB_USERNAME=root
DB_PASSWORD=
```

MEMBUAT TABEL

```
php artisan make:migration create_mahasiswa_table
```

kemudian modifikasi file migration dengan nama *_create_mahasiswa_table.php yang ada pada folder database/migrations pada fungsi **up()** menjadi seperti berikut:

```
public function up()
{
    Schema::create('mahasiswa', function (Blueprint $table) {
        $table->integer('nim_mahasiswa')->primary();
        $table->char('nama_mahasiswa',100);
        $table->year('angkatan_mahasiswa');
        $table->char('kd_prodi',5);
        $table->dateTime('created_at');
        $table->dateTime('updated_at');
        $table->timestamps();
    });
}
```

Kemudian sekarang jalankan perintah **php artisan migrate** pada cmd atau comand prompt yang sudah terarah pada proyek laravel mahasiswa_api

MEMBUAT MODEL

```
php artisan make:model Mahasiswa
```



	MODUL PRAKTIKUM	JTI
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	

Hal 4 of 12

```
app > Models > 🐘 Mahasiswa.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Mahasiswa extends Model
9  {
10     use HasFactory;
11     protected $table="mahasiswa";
12     protected $fillable = [
13         'nim_mahasiswa',
14         'nama_mahasiswa',
15         'angkatan_mahasiswa',
16         'kd_prodi'
17     ];
18 }
```

MEMBUAT CONTROLLER

```
> php artisan make:controller MahasiswaController --resource --api
```

bukalah controller MahasiswaController kemudian modifikasi fungsi store menjadi seperti berikut:



	MODUL PRAKTIKUM	JTI
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	

Hal 5 of 12

```
App > Http > Controllers > MahasiswaController.php > MahasiswaController > store
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\Mahasiswa;
7 use Facade\FlareClient\Http\Response;
8 use Illuminate\Database\QueryException;
9 use Illuminate\Support\Facades\Validator;
10 use Symfony\Component\HttpFoundation\Response as HttpFoundationResponse;
11
12 class MahasiswaController extends Controller
13 {
14     /**
15      * Display a listing of the resource.
16      *
17      * @return \Illuminate\Http\Response
18      */
19     public function index()
20     {
21         $mahasiswa = Mahasiswa::orderBy('created_at', 'DESC')->paginate(5);
22         $response = [
23             'message' => 'Data Mahasiswa',
24             'data' => $mahasiswa,
25         ];
26         return response()->json($response, HttpFoundationResponse::HTTP_OK);
27     }
28 }
```



	MODUL PRAKTIKUM	JTI
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	

Hal 6 of 12

```
public function store(Request $request)
{
    $validator = Validator::make($request->all(), [
        "nim_mahasiswa" => ['required'],
        "nama_mahasiswa" => ['required'],
        "angkatan_mahasiswa" => ['required'],
        "kd_prodi" => ['required']
    ]);

    if ($validator->fails()) {
        return response()->json(
            $validator->errors(),
            HttpFoundationResponse::HTTP_UNPROCESSABLE_ENTITY
        );
    }

    try {
        $mahasiswa = Mahasiswa::create($request->all());

        $response = [
            'message' => 'Berhasil disimpan',
            'data' => $mahasiswa,
        ];

        return response()->json($response, HttpFoundationResponse::HTTP_CREATED);
    } catch (QueryException $e) {
        return response()->json([
            'message' => "Gagal " . $e->errorInfo,
        ]);
    }
}
```

```
public function show($id)
{
    $mahasiswa = Mahasiswa::where('nim_mahasiswa', $id)->firstOrFail();
    if (is_null($mahasiswa)) {
        return $this->sendError('Mahasiswa tidak ditemukan');
    }
    return response()->json([
        "success" => true,
        "message" => "Data Mahasiswa ditemukan.",
        "data" => $mahasiswa,
    ]);
}
```

	MODUL PRAKTIKUM	JTI Hal 7 of 12
	PEMROGRAMAN WEB LANJUT	
	RESTFUL API	

```
public function update(Request $request, $id)
{
    $mahasiswa = Mahasiswa::where('nim_mahasiswa', $id);
    $mahasiswa->update($request->all());
    return response()->json([
        "success" => true,
        "message" => "Data Mahasiswa telah diubah.",
        "data" => $mahasiswa,
    ]);
}
```

```
110     public function destroy($id)
111     {
112         $deletedRows = Mahasiswa::where('nim_mahasiswa', $id)->delete();
113         return response()->json([
114             "success" => true,
115             "message" => "Data Mahasiswa berhasil dihapus.",
116             "data" => $deletedRows,
117         ]);
118     }
119 }
```

MEMBUAT ROUTE

Perbaharui file route API pada file mahasiswa_api/routes/api.php seperti dibawah ini:

```
routes > api.php > ...
1  <?php
2
3  use Illuminate\Http\Request;
4  use Illuminate\Support\Facades\Route;
5
6  use App\Http\Controllers\MahasiswaController;
7  Route::get('/mahasiswa', [MahasiswaController::class, 'index']);
8  Route::get('/mahasiswa/show/{id}', [MahasiswaController::class, 'show']);
9  Route::delete('/mahasiswa/destroy/{id}', [MahasiswaController::class, 'destroy']);
10 Route::post('/mahasiswa', [MahasiswaController::class, 'store']);
11 Route::put('/mahasiswa/update/{id}', [MahasiswaController::class, 'update']);
```

PENGUJIAN API

Download program aplikasi disini:

<https://www.postman.com/downloads/>



MODUL PRAKTIKUM PEMROGRAMAN WEB LANJUT RESTFUL API

JTI
Hal 8 of 12

Postman interface showing the 'My Workspace' overview. The 'Requests' button in the 'In this workspace' section is circled in red.

Activity Log:

- November 30, 2022
- Moh Ali Fikri added the New Environment environment 10:51 PM
- Moh Ali Fikri renamed the workspace from My Work to My Workspace 10:43 PM
- Moh Ali Fikri renamed the workspace from Mahasiswa API to My Work 10:43 PM
- Moh Ali Fikri deleted the New Environment environment 10:41 PM
- Moh Ali Fikri deleted the Mahasiswa API 10:40 PM
- Moh Ali Fikri renamed the API from New API to Mahasiswa API 10:38 PM
- Moh Ali Fikri added the New API API 10:38 PM
- Moh Ali Fikri renamed the workspace from My Workspace to Mahasiswa API 10:36 PM
- Moh Ali Fikri deleted the Mahasiswa API 10:36 PM
- November 29, 2022

Postman interface showing a GET request to `http://localhost:8000/api/mahasiswa`. The request bar and the response body (JSON) are circled in red.

Request Details:

- Method: GET
- URL: `http://localhost:8000/api/mahasiswa`
- Send button

Response Body (JSON):

```
1 {
2   "message": "Data Mahasiswa",
3   "data": {
4     "current_page": 1,
5     "data": [],
6     "first_page_url": "http://localhost:8000/api/mahasiswa?page=1",
7     "from": null,
8     "last_page": 1,
9     "last_page_url": "http://localhost:8000/api/mahasiswa?page=1",
10    "links": [
11      {
12        "url": null,
13        "label": "< Previous",
14      },
15      {
16        "url": "http://localhost:8000/api/mahasiswa?page=1",
17        "label": "1",
18      },
19      {
20        "url": "http://localhost:8000/api/mahasiswa?page=2",
21        "label": "Next >",
22      }
23    ]
24  }
25 }
```

Overview POST http://localhost:8000/ No Environment

http://localhost:8000/api/mahasiswa Save

POST **1** http://localhost:8000/api/mahasiswa **2** Send **4**

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none
 ☐ form-data
 ☒ x-www-form-urlencoded
 ☐ raw
 ☐ binary
 ☐ GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	nim_mahasiswa	22030001			
<input checked="" type="checkbox"/>	nama_mahasiswa	Adeeva 3			
<input checked="" type="checkbox"/>	angkatan_mahasiswa	2022			
<input checked="" type="checkbox"/>	kd_prodi	D3TI			
	Key	Value	Description		

Body Cookies Headers (10) Test Results 201 Created 366 ms 545 B Save Response

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "Berhasil disimpan",
3    "data": {
4      "nim_mahasiswa": "22030001",
5      "nama_mahasiswa": "Adeeva",
6      "angkatan_mahasiswa": "2022",
7      "kd_prodi": "D3TI",
8      "updated_at": "2022-12-05T02:04:53.000000Z",
9      "created_at": "2022-12-05T02:04:53.000000Z",
10     "id": 0
11   }
  
```

5

Cookies Capture requests Runner Trash

GET
1

http://localhost:8000/api/mahasiswa/show/22030001
2

Send
3

Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies

none
form-data
x-www-form-urlencoded
raw
binary
GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input type="checkbox"/>	nim_mahasiswa	22030001			
<input type="checkbox"/>	nama_mahasiswa	Adeeva			
<input type="checkbox"/>	angkatan_mahasiswa	2022			
<input type="checkbox"/>	kd_prodi	D3TI			
	Key	Value	Description		

Body
Cookies
Headers (10)
Test Results

200 OK
145 ms
552 B
Save Response

Pretty
Raw
Preview
Visualize
JSON

```

1
2
3
4
5
6
7
8
9
10
11
12
{
  "success": true,
  "message": "Data Mahasiswa ditemukan.",
  "data": {
    "nim_mahasiswa": 22030001,
    "nama_mahasiswa": "Adeeva",
    "angkatan_mahasiswa": 2022,
    "kd_prodi": "D3TI",
    "created_at": "2022-12-05T02:04:53.000000Z",
    "updated_at": "2022-12-05T02:04:53.000000Z"
  }
}

```

Cookies
Capture requests
Runner
Trash

PUT **1**

http://localhost:8000/api/mahasiswa/update/22030001 **2**

Send **4**

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input type="checkbox"/>	nim_mahasiswa	22030001			
<input checked="" type="checkbox"/>	nama_mahasiswa	Adeeva Nadia			
<input checked="" type="checkbox"/>	angkatan_mahasiswa	2022 3			
<input checked="" type="checkbox"/>	kd_prodi	D3TI			
	Key	Value	Description		

Body

Cookies

Headers (10)

Test Results

200 OK

148 ms

375 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```

1  {
2    "success": true,
3    "message": "Data Mahasiswa telah diubah.",
4    "data": {}
5  }
```

5

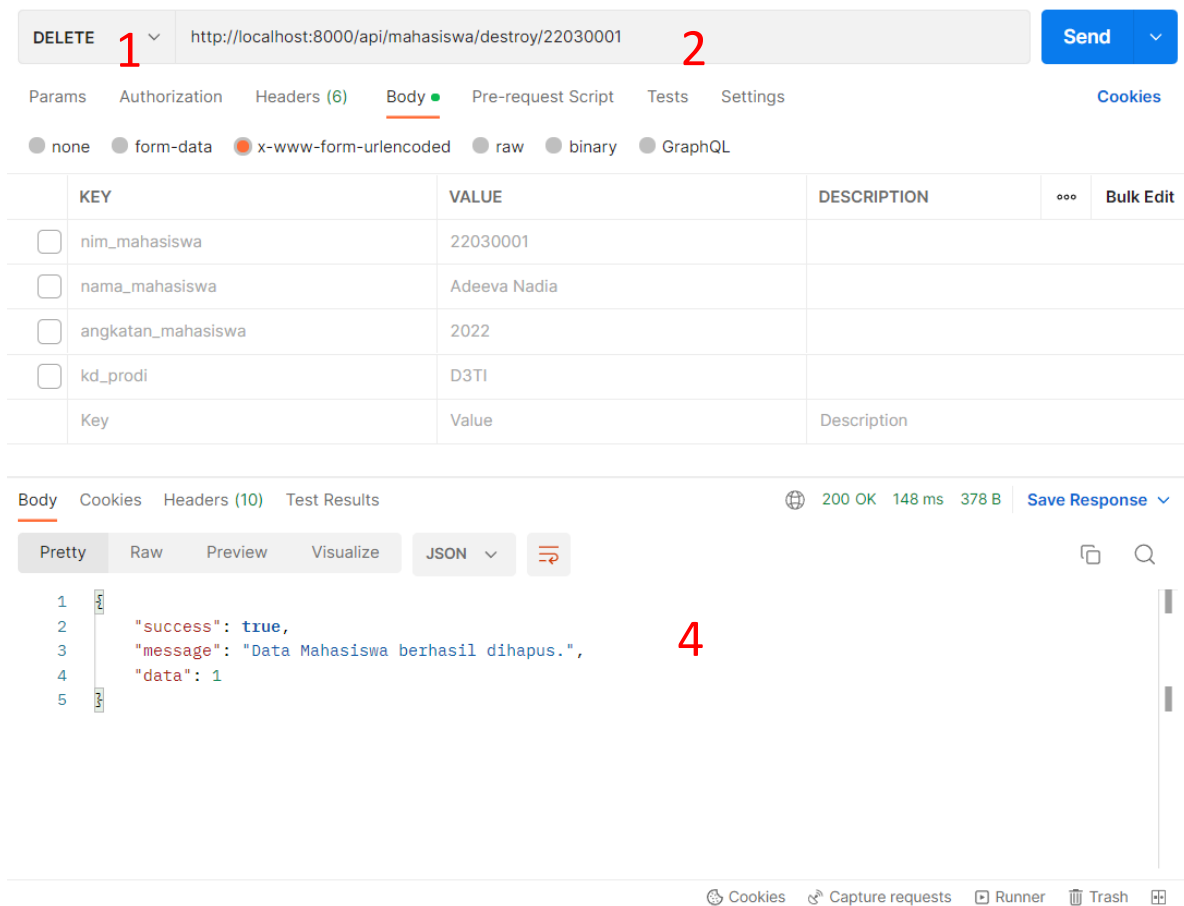
Cookies

Capture requests

Runner

Trash

1



The screenshot shows a Postman interface with a DELETE request to `http://localhost:8000/api/mahasiswa/destroy/22030001`. The request is in the 'Body' tab, and the response is in the 'Body' tab, showing a successful JSON response.

Request Details:

- Method: DELETE
- URL: `http://localhost:8000/api/mahasiswa/destroy/22030001`
- Body Type: x-www-form-urlencoded

Request Data:

KEY	VALUE	DESCRIPTION
nim_mahasiswa	22030001	
nama_mahasiswa	Adeeva Nadia	
angkatan_mahasiswa	2022	
kd_prodi	D3TI	

Response Details:

- Status: 200 OK
- Time: 148 ms
- Size: 378 B

Response Body (JSON):

```

{
  "success": true,
  "message": "Data Mahasiswa berhasil dihapus.",
  "data": 1
}
  
```

TUGAS

1. Buatlah sebuah RESTful tabel prodi untuk melakukan proses tampil, tambah data, perubahan data, dan hapus.
2. Lakukan pengujiannya menggunakan postman

Catatan:

- Langkah-langkah pembuatannya dapat mengikuti praktikum
- Laporan praktikum tugas dan sourcecode dikumpulkan ke elearning