

BAB 10

REST API

A. MATERI

Rest API

B. TUJUAN

- Mampu mempersiapkan program REST API
- Mampu melakukan praktikum dengan kinerja mandiri, bermutu, terukur, mampu bekerja sama, berkomunikasi dan mengambil inisiatif

C. Sub- CPMK

Sub – CPMK 8 : Mampu mempersiapkan program REST API dengan kinerja mandiri, bermutu, terukur, mampu bekerja sama, berkomunikasi dan mengambil inisiatif dengan kinerja mandiri, bermutu, terukur, mampu bekerja sama, berkomunikasi dan mengambil inisiatif (Minggu 9) (C3, P2, A3)

D. TEORI

10.1. REST API

API atau Application Programming merujuk pada perangkat lunak dengan fungsi yang berbeda. Kata Antarmuka dapat diartikan sebagai kontrak layanan antara dua aplikasi yang saling berkomunikasi dengan menggunakan permintaan (request) dan respons (response). Dokumentasi API keduanya berisi informasi cara developer menyusun permintaan dan respons tersebut. Aplikasi yang mengirimkan permintaan disebut sebagai klien dan aplikasi yang mengirimkan respons disebut sebagai server [7].

REST merupakan kepanjangan dari Representational State Transfer yang mendefinisikan fungsi-fungsi diantaranya GET, PUT, DELETE, dll. Fungsi – Fungsi tersebut dapat digunakan klien untuk mengakses data server. Klien dan server saling bertukar data dengan menggunakan HTTP [7].

Fitur utama API REST yaitu sifat *stateless*-nya. Bersifat *stateless* berarti server tidak menyimpan data klien di antara permintaan. Permintaan klien ke server mirip dengan URL yang Anda ketik di peramban untuk mengunjungi sebuah situs web. Respons dari server berupa data plain tanpa rendering grafis umum halaman web [7].

10.2. Laravel

Laravel adalah framework aplikasi web yang ekspresif, sintak elegan . Framework web menyediakan struktur dan titik awal untuk membuat aplikasi. Laravel memberikan fitur – fitur canggih seperti *thorough dependency injection, an expressive database abstraction layer, queues and scheduled jobs, unit and integration testing*, dan lain-lain [8].

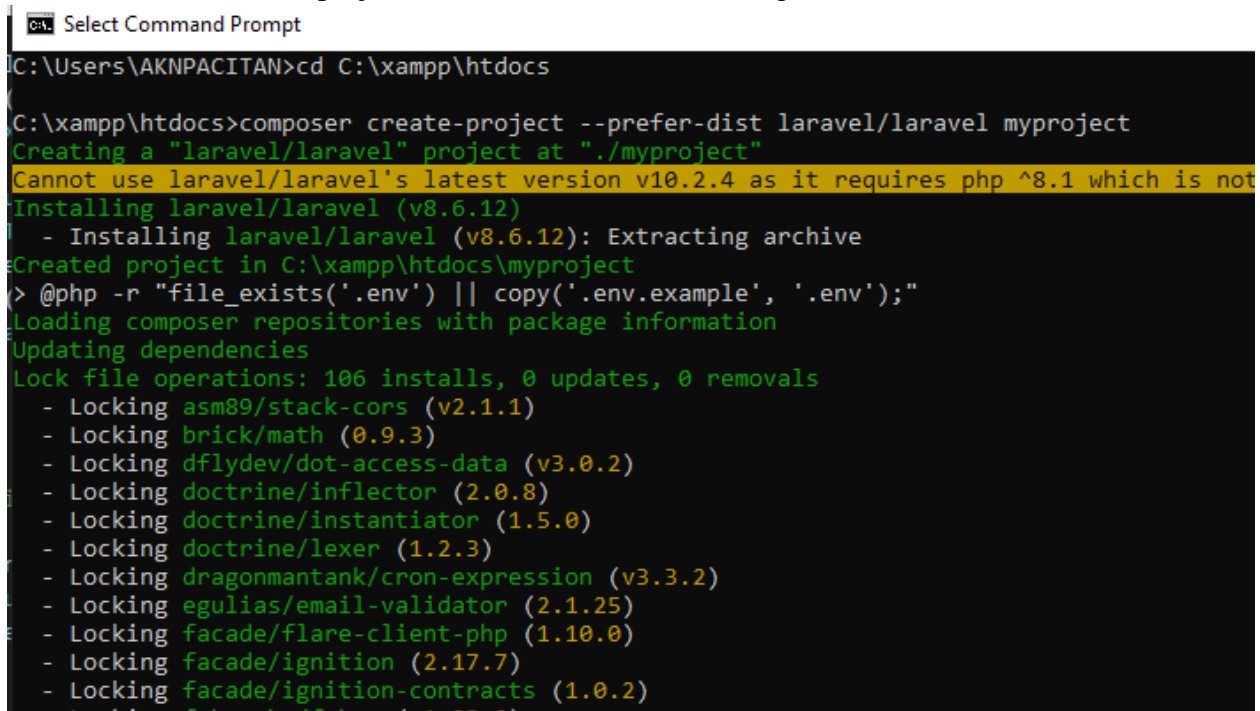
Saat membuat API, mungkin diperlukan *transformation layer* yang berada diantara *eloquent models* dan *JSON responses* yang dikembalikan ke aplikasi user. *Eloquent's resource classes* memungkinkan untuk mengubah model dan *model collections* ke format JSON menggunakan method `toJson` [8].

10.3. JSON

JSON(JavaScript Object Notation) adalah format pertukaran data yang ringan dan mudah. Objek adalah sekumpulan pasangan nama/nilai yang tidak terurut. Sebuah objek diawali dengan {kurung kurawal kiri dan diakhiri dengan }kurung kurawal kanan. Setiap nama diikuti oleh :titik dua dan pasangan nama/nilai dipisahkan oleh ,koma [9]. Format JSON dapat dipelajari lebih lanjut di <https://www.hostinger.co.id/tutorial/apa-itu-json>

10.4. Membuat REST API di Laravel

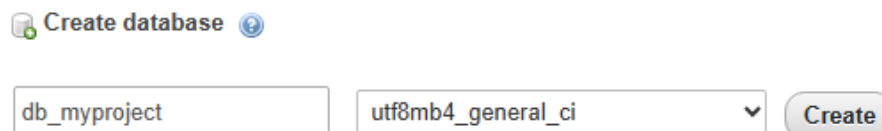
Pertama buatlah project laravel terlebih dahulu, sebagaimana Gambar 10.1.



```
C:\Users\AKNPACITAN>cd C:\xampp\htdocs
C:\xampp\htdocs>composer create-project --prefer-dist laravel/laravel myproject
Creating a "laravel/laravel" project at "./myproject"
Cannot use laravel/laravel's latest version v10.2.4 as it requires php ^8.1 which is not
Installing laravel/laravel (v8.6.12)
- Installing laravel/laravel (v8.6.12): Extracting archive
Created project in C:\xampp\htdocs\myproject
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 106 installs, 0 updates, 0 removals
- Locking asm89/stack-cors (v2.1.1)
- Locking brick/math (0.9.3)
- Locking dflydev/dot-access-data (v3.0.2)
- Locking doctrine/inflator (2.0.8)
- Locking doctrine/instantiator (1.5.0)
- Locking doctrine/lexer (1.2.3)
- Locking dragonmantank/cron-expression (v3.3.2)
- Locking egulias/email-validator (2.1.25)
- Locking facade/flare-client-php (1.10.0)
- Locking facade/ignition (2.17.7)
- Locking facade/ignition-contracts (1.0.2)
```

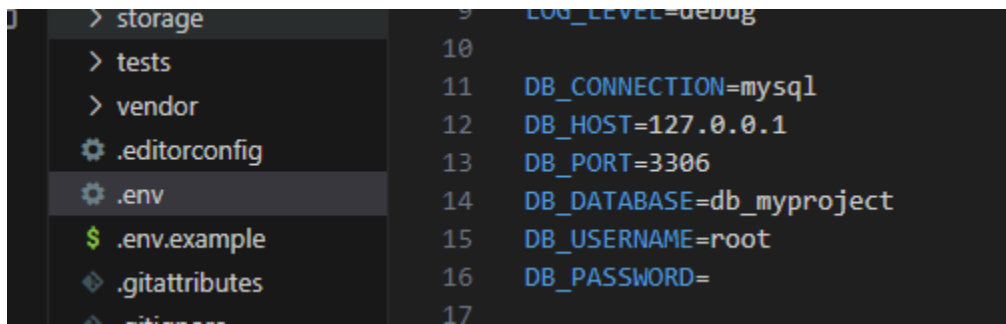
Gambar 10.1. Membuat Proyek Laravel

Kemudian buatlah database sebagaimana Gambar 10.2.



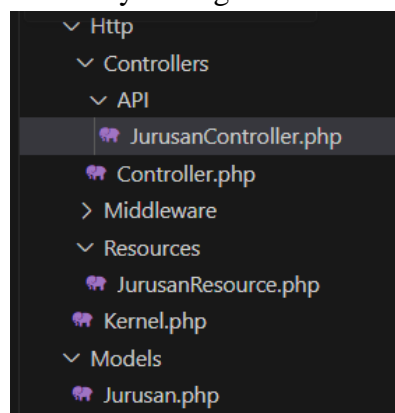
Gambar 10.2. Membuat Database

Buka project laravel yang telah dibuat di Visual Studio Code, edit file `.env` sebagaimana Gambar 10.3.. Setting nama database sebagaimana nama database yang dibuat pada Gambar 10.2.



Gambar 10.3. Setting File .env

Struktur file yang akan kita nantinya sebagaimana Gambar 10.4. berikut



Gambar 10.4. Struktur File pada Project

Selanjutnya membuat model dan migration dari tabel Jurusan. Langkah ikuti sebagaimana Gambar 10.5. dan 10.6. Contoh kasus pada praktikum ini adalah tabel Jurusan yang memiliki field *idJurusan* sebagai *primary key* dan *namaJurusan*. *idJurusan* di atur sebagai *auto increment* agar saat input data nilainya otomatis terinput secara terurut.

```
C:\xampp\htdocs\myproject>php artisan make:model Jurusan -m
Model created successfully.
Created Migration: 2023_07_10_041854_create_jurusans_table
C:\xampp\htdocs\myproject>
```

Gambar 10.5. Membuat Model dan Migration tabel Jurusan

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
```

```

{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('jurusans', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('jurusans');
    }
};

```

Gambar 10.6. Migration Tabel Jurusan

```

1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Jurusan extends Model
9  {
10     use HasFactory;
11
12     protected $fillable=['name'];
13 }

```

Gambar 10.7. Konfigurasi Model Jurusan

Jika *primary key* dari tabel Jurusan bukan *id* , maka perlu dijelaskan bahwa *primary key* tersebut. *fillable* adalah *field* yang harus diisi. Kemudian ketikkan perintah *php artisan migrate* sebagaimana Gambar 10.8, maka akan termigrate file-file migration, salah satunya *migration* dari

tabel jurusan dan terbentuk tabel jurusan pada database yang sebelumnya dibuat sebagaimana Gambar 10.9

```
C:\xampp\htdocs\myproject>php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (28.98ms)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (42.34ms)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (29.98ms)
Migrating: 2019_12_14_000001_create_personal_access_tokens_table
Migrated: 2019_12_14_000001_create_personal_access_tokens_table (40.47ms)
Migrating: 2023_07_10_041854_create_jurusans_table
Migrated: 2023_07_10_041854_create_jurusans_table (22.91ms)
C:\xampp\htdocs\myproject>
```

Gambar 10.7. Menjalankan *php artisan migrate*

#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default	Cc
1	id	BIGINT	20	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREME...	
2	name	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default	
3	created_at	TIMESTAMP		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
4	updated_at	TIMESTAMP		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	

Gambar 10.8. Hasil Migration Tabel Jurusan

Kemudian buat *JurusanResource* sebagaimana Gambar 10.9.

```
C:\xampp\htdocs\myproject>php artisan make:resource JurusanResource
Resource created successfully.
```

Gambar 10.9. Membuat *JurusanResource*

Isi file tersebut sebagaimana program dibawah ini. Setiap resource class mendefinisikan method toArray yang me-return attribute array yang harus dikonversi ke JSON ketika resource dikembalikan sebagai response dari route atau controller.

```
namespace App\Http\Resources;

use Illuminate\Http\Request;
use Illuminate\Http\Resources\Json\JsonResource;

class JurusanResource extends JsonResource
{
    /**
     * Transform the resource into an array.
     *
     * @return array<string, mixed>
     */
    public function toArray(Request $request): array
```

```

{
    return [
        'idJurusan'=>$this->id,
        'namaJurusan'=>$this->name
    ];
}
}

```

Kemudian membuat controller dalam directory *app/Http/Controllers/Api*

php artisan make:controller Api/JurusanController --resource --api

Akan muncul JurusanController lengkap bersama fungsi-fungsinya kecuali create() dan edit karena tidak dibutuhkan pada REST API.

```

<?php

namespace App\Http\Controllers\Api;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;

class JurusanController extends Controller
{
    /**
     * Display a listing of the resource.
     */
    public function index()
    {

    }

    /**
     * Store a newly created resource in storage.
     */
    public function store(Request $request)
    {

    }

    /**
     * Display the specified resource.
     */
    public function show(Jurusan $jurusan)

```

```

{
}

/**
 * Update the specified resource in storage.
 */
public function update(Request $request, Jurusan $jurusan)
{
}

/**
 * Remove the specified resource from storage.
 */
public function destroy(Jurusan $jurusan)
{
}
}

```

Langkah selanjutnya membuat REST API Routes pada file *routes/api.php*. Kemudian lakukan pengecekan *route* yang telah terbuat. Seperti Gambar 10.11.

```
Route::resource('programs', App\Http\Controllers\API\ProgramController::class);
```

PS C:\xampp\htdocs\myproject> php artisan route:list

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	/		Closure	web
	GET HEAD	api/jurusans	jurusans.index	App\Http\Controllers\API\JurusanController@index	api
	POST	api/jurusans	jurusans.store	App\Http\Controllers\API\JurusanController@store	api
	GET HEAD	api/jurusans/create	jurusans.create	App\Http\Controllers\API\JurusanController@create	api
	GET HEAD	api/jurusans/{jurusan}	jurusans.show	App\Http\Controllers\API\JurusanController@show	api
	PUT PATCH	api/jurusans/{jurusan}	jurusans.update	App\Http\Controllers\API\JurusanController@update	api
	DELETE	api/jurusans/{jurusan}	jurusans.destroy	App\Http\Controllers\API\JurusanController@destroy	api
	GET HEAD	api/jurusans/{jurusan}/edit	jurusans.edit	App\Http\Controllers\API\JurusanController@edit	api
	GET HEAD	api/user		Closure	api
	GET HEAD	sanctum/csrf-cookie		Laravel\Sanctum\Http\Controllers\CsrfCookieController@show	App\Http\Middleware\Authenticate:sanctum web

PS C:\xampp\htdocs\myproject>

Gambar 10.11. List Route

Install Postman. Postman digunakan untuk membangun dan menggunakan API. Download di <https://www.postman.com/downloads/>

Install ngrok di <https://ngrok.com/download>

1. Unzip to install

On Linux or Mac OS X you can unzip ngrok from a terminal with the following command. On Windows, just double click ngrok.zip to extract it.

2. Connect your account

Running this command will add your authtoken to the default `ngrok.yml` configuration file. This will grant you access to more features and longer session times. Running tunnels will be listed on the [endpoints page](#) of the dashboard.

```
ngrok config add-authtoken 2NH41TNHYi4Ql0Dk84ufq9JgM1C_2XQTAaiHkV8iMvKhk8RUp
```

3. Fire it up

Read [the documentation](#) on how to use ngrok. Try it out by running it from the command line:

```
ngrok help
```

To start a HTTP tunnel forwarding to your local port 80, run this next:

```
ngrok http 80
```

10.4.1. Membuat REST API untuk mendapatkan data Jurusan

Edit `JurusanController` -> `index()`

```
public function index()
{
    return JurusanResource::collection(Jurusan::all());
}
```

Coba untuk mendapatkan data Jurusan dengan method GET

The screenshot shows a REST client interface with the following details:

- URL: `http://127.0.0.1:8000/api/jurusan`
- Method: `GET`
- Response Status: `200 OK`, `207 ms`, `426 B`
- Response Body (JSON):

```
{
  "data": [
    {
      "idJurusan": 1,
      "namaJurusan": "PKJ"
    },
    {
      "idJurusan": 2,
      "namaJurusan": "TSP"
    }
  ]
}
```

10.4.2. Membuat REST API untuk create data Jurusan

Edit `JurusanController` pada fungsi `store()`

```
public function store(Request $request)
```



```

{
  $validated = $request->validate([
    'name' => 'required',
  ]);

  return new JurusanResource(Jurusan::create($validated));
}

```

Coba lakukan insert dari POSTMAN

History: Today, POST http://127.0.0.1:8000/api/jurusan

POST http://127.0.0.1:8000/api/jurusan

Params: Authorization: Headers (8): Body (selected) | Pre-request Script: Tests: Settings: Cookies

Body format: none, form-data, x-www-form-urlencoded (selected), raw, binary

Key	Value	Bulk Edit
<input checked="" type="checkbox"/> name	Pemeliharaan Komputer dan Jaringan	
Key	Value	

Body: Cookies: Headers (10): Test Results: 201 Created 239 ms 388 B Save Response

Body format: Pretty (selected), Raw, Preview, Visualize, JSON

```

1  {
2    "data": {
3      "idJurusan": 4,
4      "namaJurusan": "Pemeliharaan Komputer dan Jaringan"
5    }
6  }

```

Console: Not connected to a Postman account

POST http://127.0.0.1:8000/api/jurusan Send

Params: Authorization: Headers (11) (selected) | Body: Pre-request Script: Tests: Settings: Cookies

<input checked="" type="checkbox"/>	Content-Length	<calculated when request is sent>
<input checked="" type="checkbox"/>	Host	<calculated when request is sent>
<input checked="" type="checkbox"/>	User-Agent	PostmanRuntime/7.33.0
<input checked="" type="checkbox"/>	Accept	*
<input checked="" type="checkbox"/>	Accept-Encoding	gzip, deflate, br
<input checked="" type="checkbox"/>	Connection	keep-alive
<input checked="" type="checkbox"/>	Accept	application/json

HTTP <http://127.0.0.1:8000/api/jurusan> Save

POST <http://127.0.0.1:8000/api/jurusan> Send

Params Authorization Headers (11) **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"/>	name				
	Key	Value	Description		

Body Cookies (2) Headers (10) Test Results 422 Unprocessable Content 255 ms 418 B Save as example

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "The name field is required.",
3    "errors": {
4      "name": [
5        "The name field is required."
6      ]
7    }
8  }

```

10.4.3. Membuat REST API untuk update data Jurusan

Membuat UpdateJurusanRequest

```
PS C:\laragon\www\rest_api> php artisan make:request UpdateJurusanRequest
```

Edit File UpdateJurusanRequest

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class UpdateJurusanRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }
}
```

```

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string,
\Illuminate\Contracts\Validation\ValidationRule|array<mixed>|string>
     */
    public function rules(): array
    {
        return [
            'name' => 'required'
        ];
    }
}

```


Edit JurusanController

```

public function update(UpdateJurusanRequest $request, Jurusan $jurusan)
{
    $jurusan->update($request->validated());
    return new JurusanResource($jurusan);
}

```

Coba Edit Jurusan melalui postman

 **http://127.0.0.1:8000/api/jurusan/3**

GET

http://127.0.0.1:8000/api/jurusan/3

Send

ParamsAuthorizationHeaders (11)BodyPre-request ScriptTestsSettingsCookies

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


	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	name	PKJ			
	Key	Value	Description		

BodyCookies (2)Headers (10)Test Results

200 OK233 ms352 BSave as example

PrettyRawPreviewVisualizeJSON

```
1
2  "data": {
3    "idJurusan": 3,
4    "namaJurusan": "PKJ"
5  }
6
```

 **http://127.0.0.1:8000/api/jurusan/3**

PUT

http://127.0.0.1:8000/api/jurusan/3

Send

ParamsAuthorizationHeaders (11)BodyPre-request ScriptTestsSettingsCookies

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

	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	name	Pemeliharaan Komputer dan Jaringan			
	Key	Value	Description		

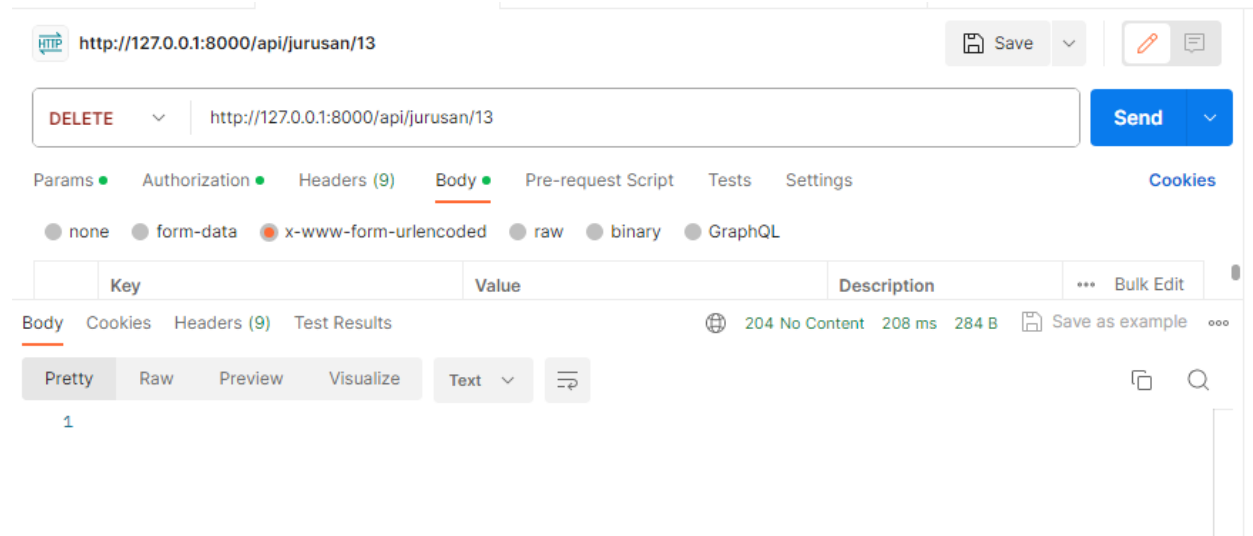
BodyCookies (2)Headers (10)Test Results

200 OK210 ms383 BSave as example

PrettyRawPreviewVisualizeJSON

```
1
2  "data": {
3    "idJurusan": 3,
4    "namaJurusan": "Pemeliharaan Komputer dan Jaringan"
5  }
6
```

10.4.4. Membuat REST API untuk Delete data Jurusan



Register dan Login

```
composer require laravel/sanctum
```

```
php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"
```

```
php artisan migrate
```

```
app > Models > Jurusan.php > PHP Intelephense > Jurusan
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7  use Illuminate\Notifications\Notifiable;
8  use Laravel\Sanctum\HasApiTokens;
9
10 class Jurusan extends Model
11 {
12     use HasApiTokens, HasFactory, Notifiable;
13
14     protected $fillable=['name'];
15 }
16
```

edit api.php

```
Route::group(['middleware' => 'auth:sanctum'], function () {
    Route::resource('jurusan', JurusanController::class)->only([
        'store', 'update', 'index',
    ])
```

```
        'destroy'
    ]);
});
```

php artisan make:controller Api/AuthController

```
Route::post('/auth/login', [AuthController::class, 'login']);
Route::post('/auth/register', [AuthController::class, 'register']);
```

```
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules\Password;
use Illuminate\Validation\ValidationException;

class AuthController extends Controller
{
    public function register(Request $request)
    {
        $request->validate([
            'name' => 'required|string',
            'email' => 'required|string|email|unique:users',
            'password' => [
                'required', 'string', 'min:8',
                'confirmed', Password::defaults()
            ],
            'device_name' => 'required',
        ]);
        $user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
        ]);
        // return $user->createToken($request->device_name)->plainTextToken;
        return response()->json(
            $user->createToken($request->device_name)->plainTextToken
        );
    }
}
```

```

public function login(Request $request)
{
    $request->validate([
        'email' => 'required|email',
        'password' => 'required',
        // 'device_name' => 'required',
    ]);
    $user = User::where('email', $request->email)->first();
    if (!$user || !Hash::check($request->password, $user->password)) {
        throw ValidationException::withMessages([
            'email' => ['The provided credentials are
incorrect.'],
        ]);
    }
    // return $user->createToken($request->device_name)->plainTextToken;
    return response()->json(
        $user->createToken($request->device_name)->plainTextToken
    );
}
}

```

HTTP <http://127.0.0.1:8000/api/jurusan/?name=test> Save

POST <http://127.0.0.1:8000/api/jurusan/?name=test> Send

Params ● Authorization Headers (10) Body ● Pre-request Script Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	name	test			
	Key	Value	Description		

Body Cookies Headers (8) Test Results 401 Unauthorized 159 ms 298 B Save as example ...

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "Unauthenticated."
3  }

```

Register : output masih salah coba lagi



http://127.0.0.1:8000/api/auth/register

Save



POST



http://127.0.0.1:8000/api/auth/register

Send



Params Authorization Headers (11) **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"/>	email	berlianlili@aknpacitan.ac.id			
<input checked="" type="checkbox"/>	password	123AKNPacitan			
<input checked="" type="checkbox"/>	name	berlian			
<input checked="" type="checkbox"/>	password_confirmation	123AKNPacitan			
<input checked="" type="checkbox"/>	device_name	123			

Body Cookies Headers (10) Test Results

200 OK 297 ms 362 B Save as example

Pretty

Raw

Preview

Visualize

JSON



1 "131|V0Zd2BreAeIlhZ7UE0sfe5BGzT4bH5o1gKa80t1vd435ef66"



http://127.0.0.1:8000/api/auth/login

Save



POST



http://127.0.0.1:8000/api/auth/login

Send



Params Authorization Headers (11) **Body** Pre-request Script Tests Settings

Cookies

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

<input type="checkbox"/>	password_confirmation	123AKNPacitan	
<input checked="" type="checkbox"/>	device_name	123	
	Key	Value	Description

Body Cookies Headers (10) Test Results

200 OK 302 ms 360 B Save as example

Pretty

Raw

Preview

Visualize

JSON



1 "7|EARYeEZu2vK6sXWlJ3tCtMwnRGYt8hbhGxQ7BAun26d6984f"

Params

Authorization

Headers (11)

Body

Pre-request Script

Tests

Settings

Cookies

Type

Bearer ...

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#)

The authorization header will be automatically generated when you send the request. Learn more about [authorization](#)

Token

2|R4qy0wQilZoAIPf7SUBXmE3yTl1Cc5it7LiZ8rLH8a612068|

HTTP

http://127.0.0.1:8000/api/jurusan

Save

POST

http://127.0.0.1:8000/api/jurusan

Send

Params

Authorization

Headers (11)

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"/>	email	berlianjuli@aknpacitan.ac.id			
<input type="checkbox"/>	password	123AKNPacitan			
<input checked="" type="checkbox"/>	name	berlian			
<input type="checkbox"/>	password_confirmation	123AKNPacitan			
<input type="checkbox"/>	device_name	123			
	Key	Value	Description		

Body

Cookies

Headers (10)

Test Results

201 Created

263 ms

362 B

Save as example

Pretty

Raw

Preview

Visualize

JSON

```
1  {
2    "data": {
3      "idJurusan": 14,
4      "namaJurusan": "berlian"
5    }
6  }
```



http://127.0.0.1:8000/api/jurusan

Save



POST



http://127.0.0.1:8000/api/jurusan

Send



Params

Authorization

Headers (10)

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"/>	email	berlianjuli@aknpacitan.ac.id			
<input type="checkbox"/>	password	123AKNPacitan			
<input checked="" type="checkbox"/>	name	berlian			
<input type="checkbox"/>	password_confirmation	123AKNPacitan			
<input type="checkbox"/>	device_name	123			
	Key	Value	Description		

Body Cookies Headers (8) Test Results



401 Unauthorized

151 ms

298 B



Save as example



Pretty

Raw

Preview

Visualize

JSON



```
1 {
2   "message": "Unauthenticated."
3 }
```

Route :

```
Route::group(['middleware' => 'auth:sanctum'], function () {  
    Route::post('/auth/logout', [AuthController::class, 'logout']);  
});
```

AuthController :

```
public function logout(Request $request)  
{  
    $request->user()->currentAccessToken()->delete();  
    return response()->noContent();  
}
```