

# **Laporan Ujian Akhir Semester**

## **Laporan Hasil Pratikum Interperability**

### **Dibidang Software Development**

**Laporan Hasil Pratikum untuk memenuhi nilai Ujian Akhir Semester pada mata  
kuliah Pratikum Interperability**



Disusun Oleh:

**Arya\_Rizki\_Pratama** D112011009  
**Ahmad Febriyanto** D112011002

**Program Study Teknik Informatika**

**Fakultas Teknik**

**Politeknik TEDC Bandung 2023**

## **CHAPTER ONE**

### **RINGKASAN**

RESTful API / REST API merupakan penerapan dari API (Application Programming Interface). REST (Representational State Transfer) adalah sebuah arsitektur metode komunikasi yang menggunakan protokol HTTP untuk pertukaran data dimana metode ini sering diterapkan dalam pengembangan aplikasi. API secara eksplisit memanfaatkan metodologi HTTP yang ditentukan oleh protokol RFC 2616. Permintaan API dapat menggunakan GET untuk mengambil data, PUT untuk mengubah atau memperbarui data, POST untuk membuat data, dan DELETE untuk menghapus data. REST API memiliki empat komponen penting diantaranya adalah URL Design, HTTP Verbs, HTTP Response Code, dan Format Response.

Fitur merupakan unsur penting dalam pembuatan suatu produk. Fitur merupakan sarana kompetitif untuk mendiferensiasikan atau membedakan produk perusahaan dengan produk pesaing. Service App yang telah kami bangun merupakan jawaban terhadap kebutuhan user. Beberapa fitur yang dibutuhkan diantaranya adalah fitur input data (hewan, pemilik dan dokter), fitur update data (hewan, pemilik dan dokter) fitur hapus data (hewan, pemilik dan dokter), dan juga fitur untuk menampilkan data yang telah dimasukan sebelumnya baik secara satu per satu maupun tampilan secara keseluruhan data. Menjawab atas apa yang dibutuhkan oleh user, maka kami mengembangkan fitur CRUD-in aja ( Create Read Update dan Delete ). Fitur ini nantinya akan memudahkan para petugas klinik untuk meakukan pencatatan terhadap data pasien / client mereka.

Endpoint API adalah titik di mana API kode yang memungkinkan dua program perangkat lunak untuk berkomunikasi satu sama lain terhubung dengan program perangkat lunak. Dengan kata lain, Endpoint API adalah lokasi digital spesifik tempat permintaan informasi dikirim oleh program untuk mengambil sumber daya digital yang ada di sana. Lantas bagaimanakah Endpoint API bekerja?. Sistem yang berkomunikasi melalui API adalah sistem terintegrasi. Satu sisi mengirimkan informasi ke API dan disebut server. Agar permintaan yang efektif diproses oleh Endpoint, klien harus menyediakan pencari sumber daya seragam (URL), metode, daftar header, dan badan.

## **CHAPTER TWO**

### **PEMBUKAAN**

RESTful API / REST API merupakan penerapan dari API (Application Programming Interface). Sedangkan REST (Representational State Transfer) adalah sebuah arsitektur metode komunikasi yang menggunakan protokol HTTP untuk pertukaran data dimana metode ini sering diterapkan dalam pengembangan aplikasi. Dengan tujuannya untuk menjadikan sistem memiliki performa yang baik, cepat dan mudah untuk di kembangkan (scale) terutama dalam pertukaran dan komunikasi data.

API secara eksplisit memanfaatkan metodologi HTTP yang ditentukan oleh protokol RFC 2616. Permintaan ke API bisa menggunakan GET untuk mengambil sumber daya, PUT untuk mengubah status atau memperbarui sumber daya, yang dapat berupa objek, file, atau blok, POST untuk membuat sumber daya itu, dan DELETE untuk menghapusnya. RESTful API memiliki empat komponen penting yang saling melengkapi. Untuk lebih jelasnya, dapat dilihat dalam penjelasan berikut ini ;

#### **URL Design**

RESTful API diakses menggunakan protokol HTTP. Penamaan dan struktur URL yang konsisten akan menghasilkan API yang baik dan mudah untuk dimengerti developer. URL API biasa disebut endpoint dalam pemanggilannya.

#### **HTTP Verbs**

Setiap request yang dilakukan terdapat metode yang dipakai agar server mengerti apa yang sedang di request client:

##### **GET**

GET adalah metode HTTP Request yang paling mudah, metode ini digunakan untuk membaca atau mendapatkan data dari sumber.

##### **POST**

POST adalah metode HTTP Request yang digunakan untuk membuat data baru dengan menyisipkan data dalam body saat request dilakukan.

##### **PUT**

PUT adalah metode HTTP Request yang biasanya digunakan untuk melakukan update data resource.

### DELETE

DELETE adalah metode HTTP Request yang digunakan untuk menghapus suatu data pada resource.

### HTTP Response Code

HTTP Response Code adalah kode standarisasi dalam menginformasikan hasil request kepada client. Secara umum terdapat 3 kelompok yang biasa kita jumpai pada RESTful API yaitu :

2XX : adalah response code yang menampilkan bahwa request berhasil.

4XX : adalah response code yang menampilkan bahwa request mengalami kesalahan pada sisi client.

5XX : adalah response code yang menampilkan bahwa request mengalami kesalahan pada sisi server.

### Format Response

Setiap request yang dilakukan client akan menerima data response dari server, response tersebut biasanya berupa data XML ataupun JSON. Setelah mendapatkan data response tersebut barulah client bisa menggunakannya dengan cara memarsing data tersebut dan diolah sesuai kebutuhan.

## CHAPTER THREE

### FITUR

#### Get to Know About Features

Fitur merupakan unsur penting dalam pembuatan suatu produk. Menurut Tjiptono (2001:103) fitur adalah unsur-unsur produk yang dipandang penting oleh konsumen dan dijadikan sebagai dasar pengambilan keputusan. Fitur merupakan sarana kompetitif untuk mendiferensiasikan atau membedakan produk perusahaan dengan produk pesaing. Atas dasar pengertian tersebut, maka fitur menjadi tolak ukur pengambilan keputusan dari konsumen terhadap produk yang sedang ditawarkan. Fitur yang sesuai dengan kebutuhan konsumen atau dalam hal ini user akan menjadi pilihan bagi mereka. Maka dari itu, pemenuhan kebutuhan atau *needs* dari konsumen sangat perlu untuk diperhatikan.

Sejalan dengan hal itu, Service App yang telah kami bangun merupakan jawaban terhadap kebutuhan user. Setelah melakukan riset terhadap beberapa dokter hewan atau biasa dikenal dengan puskesmas, kami mendapatkan kesimpulan dan berusaha meng-aplikasikannya kedalam fitur Service App yang kami kembangkan. Beberapa fitur yang dibutuhkan diantaranya adalah fitur input data ( hewan, pemilik dan dokter ), fitur update data ( hewan, pemilik dan dokter ) fitur hapus data ( hewan, pemilik dan dokter ), dan juga fitur untuk menampilkan data yang telah dimasukan sebelumnya baik secara satu per satu maupun tampilan secara keseluruhan data.

### **What We Develop About**

Menjawab atas apa yang dibutuhkan oleh user, maka kami mengembangkan fitur CRUD-in aja ( Create Read Update dan Delete ). Fitur ini nantinya akan memudahkan para petugas klinik untuk melakukan pencatatan terhadap data pasien / client mereka. Adapun fitur yang diterapkan sebagai berikut:

#### **Create Data**

Memiliki fungsi untuk menambahkan atau membuat data pasien baru dengan menambahkan beberapa keterangan yang akan di bahas lebih lanjut pada Rancangan Endpoint.

#### **Read Data**

Setelah memasukan data, petugas dapat memeriksa ( read ) data yang telah diinput untuk memastikan apakah ada kesalahan dalam data atau tidak. Jika ada kesalahan maka dapat di lakukan pengeditan lanjutan pada point Update data.

#### **Update Data**

Memiliki fungsi untuk memperbaiki atau melakukan update data bila diperlukan atau bila terjadi kesalahan dalam proses input. Fitur update ini tentunya akan mempermudah dalam proses perbaikan data nantinya.

#### **Delete Data**

Fitur selanjutnya merupakan fitur yang berguna untuk menghapus data pasien nantinya. Jika terjadi kesalahan yang cukup banyak, maka petugas dapat melakukan Delete data secara langsung.

#### **User Authentication**

Fitur terakhir yang dikembangkan merupakan fitur proteksi pada database. Hanya user yang telah terdaftar yang bisa mengakses dan melakukan penngubahan pada data. Fitur

ini merupakan fitur login yang tentunya sudah umum digunakan tetapi sangat bermanfaat untuk menjaga keamanan database dari pihak yang tidak diinginkan.

## CHAPTER FOUR

### RANCANGAN ENDPOINT

#### Get to Know About Endpoint

Endpoint API adalah titik di mana API kode yang memungkinkan dua program perangkat lunak untuk berkomunikasi satu sama lain terhubung dengan program perangkat lunak. API bekerja dengan mengirimkan permintaan informasi dari aplikasi web atau server web dan menerima tanggapan.

Dengan kata lain, Endpoint API adalah lokasi digital spesifik tempat permintaan informasi dikirim oleh program untuk mengambil sumber daya digital yang ada di sana. Titik akhir menentukan di mana API dapat mengakses sumber daya dan membantu menjamin berfungsinya perangkat lunak yang digabungkan dengan benar. Performa API bergantung pada kapasitasnya untuk berhasil berkomunikasi dengan titik akhir API. Program perangkat lunak biasanya memiliki beberapa Endpoint API. Misalnya, Endpoint Instagram mencakup salah satu yang memungkinkan bisnis dan pembuat konten mengukur interaksi media dan profil, berikutnya yang memungkinkan mereka memoderasi komentar dan balasan mereka, dan yang ketiga memungkinkan mereka untuk menemukan media yang diberi tanda pagar ( *hashtag* ).

#### How does Endpoint Work?

Lantas bagaimanakah Endpoint API bekerja?. Sistem yang berkomunikasi melalui API adalah sistem terintegrasi. Satu sisi mengirimkan informasi ke API dan disebut server. Sisi lain, klien, membuat permintaan dan memanipulasi API. Dan sisi server yang menyediakan informasi yang diminta, atau sumber daya, adalah titik akhir API.

Agar permintaan yang efektif diproses oleh Endpoint, klien harus menyediakan pencari sumber daya seragam (URL), metode, daftar header, dan badan. Header menyediakan metadata tentang permintaan dan badan menyimpan data yang dikirim oleh klien ke server. Endpoint bekerja bersama-sama dengan metode API. Metode adalah permintaan yang diizinkan yang dapat dibuat, seperti GET, DELETE, PUT atau POST. Metode tersebut sering disebut kata kerja. Dalam sintaks komunikasi sering ditempatkan tepat sebelum Endpoint yang ditentukan dalam URL lengkap.

Berikut merupakan contoh penggunaan dari Endpoint API menggunakan kode yang digunakan dalam mengajukan permintaan untuk halaman statistik tertentu di situs web NBA.

GET `https://stats.nba.com/stats/allstarballotpredictor`

Dalam contoh di atas, GET adalah metodenya sedangkan endpoint adalah bagian spesifik dari alamat web yang dicatat sebagai `/stats/allstarballotpredictor`.

### Endpoint of Our Project

Setelah cukup mengenal apa itu endpoint dan bagaimana penggunaannya, berikut kami sajikan endpoint terhadap project yang kami kerjakan dalam bentuk tabel.

Method (HTTP Req)	Endpoint	Description	Controller Function
GET	/menus	Get list of all post	index
GET	/menus/{id}	Get single post	show
POST	/menus	Create new post / data	create
PUT	/menus/{id}	Update single post / data	update
DELETE	/menus/{id}	Delete single post / data	destroy

Table 1

### PetController.php

Untuk lebih jelasnya, berikut merupakan hasil implementasi yang telah kami terapkan dalam project yang kami tangani. Source code berikut merupakan source code yang terdapat dalam file MenuController.php.



```
<?php
namespace App\Http\Controllers;

use App\Models\Menu;
use Illuminate\Http\Request;

class MenuController extends Controller
{
    public function
index()
    {
        $menus = Menu::OrderBy("id", "DESC")->paginate(10);
        $outPut = [
            "message" => "menus",
            "results" => $menus
        ];
        return response()->json($menus,
200);
    }
}
```

```

//create data      public function
store(Request $request)    {
    $input = $request->all();
    $menu = Menu::create($input);
    return response()->json($menu,
200);
}

//menampilkan data per id/ read detail
public function show($id)
{
    $menu = Menu::find($id);
    if (!$menu) {
abort(404);
    }        return response()-
>json($menu, 200);
    }

    ///fungsi update data      public function
update(Request $request, $id)
{
    $input = $request->all();

    $menu = Menu::find($id);
    if (!$menu) {
abort(404);
    }

    $menu->fill($input);
    $menu->save();
    return response()-
>json($menu,200);
    }

    ///fungsi delete data
public function destroy($id)
{
    $menu = Menu::find($id);
    if (!$menu) {
abort(404);
    }

    $menu->delete();
    $message = ['message' => 'deleted successfully', '$menu_id' => $id];

```

```

        return response()->json($message,200);
    }
}

```

Source code di atas terbagi menjadi 5 function yang masing-masing nya akan memuat method dan juga ketentuan dalam mengisi data nantinya. Tentunya ini juga berhubungan dengan endpoint pada file web.php yang akan dibahas nanti. Sebelum masuk ke dalam pembahasan end point pada file web.php, kami akan membedah source code diatas terlebih dahulu. Berikut merupakan penjelasannya.

```

class PetController extends Controller
{
    public function index()
    {
        $pet = Pet::all();
        return response()->json($pet);
    }
}

```

Function pertama merupakan fuction index yang nantinya akan berguna untuk mendapatkan seluruh data yang telah di input sebelumnya. Function index ini berkaitan dengan method GET.

`$menus = Menu::all();` berfungsi untuk memanggil data dari table pet, sedangkan `return response()->json($menus);` akan mengembalikan response dalam format json.

```

public function create(Request $request)
{
    $this->validate($request, [
        "checkup_id" => "required|unique:menus",
        "kode_makanan" => "required",
        "kode_minuman" => "required",
        "cemilan" => "required",
        "harga" => "required",
        "deskripsi" => "required",
    ]);

    $data = $request->all();
}

```

```

        $menus = Menu::create($data);

        return response()->json($menus);
    }

```

Function berikutnya merupakan function create yang nantinya akan berguna untuk melakukan POST terhadap inputan data dari user ( dalam hal ini petugas klinik ). Didalam function Create terdapat field yang harus di isi dengan data, dan juga memiliki ketentuan dimana checkup\_id harus bersifat unique atau dalam kata lain checkup\_id hanya dapat digunakan oleh satu data saja untuk menghindari double update pada database. Untuk pet\_name hingga doctor\_name dibuat “required” atau wajib diisi. `$data = $request->all` berfungsi untuk mengirim request dan `$menus = Menu::create $data` berfungsi untuk mengirim atau membuat data ke dalam tabel pet, kemudian dilanjutkan dengan system yang akan mengirim return response dalam format json di baris terakhir.

```

public function show($id)
{
    $menus = Menu::find($id);           if
(! $menus) {
        return response()->json(['message' => 'Data not found, Please
check the ID!'], 404);
    }
    return response()->json($menus);
}

```

Berikutnya merupakan function show atau merupakan method GET by id yang nantinya user akan dapat melakukan pencarian data pasien / client dengan memasukan id di dalam url. `$menus = Menu::find $id` berfungsi untuk mencari data pada tabel pet berdasarkan id yang di cantumkan user. Kemudian pada source code `return response -`  
`>json 'message' => 'Data not found, Please check the ID!' 404` berfungsi untuk memberikan return response berupa message atau pesan bila nantinya id yang di masukan tidak sesuai atau belum terdaftar pada database. Terakhir, `return response >json $menus` merupakan respon yang akan dikirimkan bila berhasil untuk mendapatkan data yang di maksud oleh user.

```

public function update(Request $request, $id)

```

```

{
    $menus = Menu::find($id);
if (!$pet) {
    return response()->json(['message' => 'Data not found, Please check the ID!'], 404);
}

$this->validate($request, [
    "checkup_id" => "required|unique:menus",
    " kode_makanan" => "required",
    " kode_minuman" => "required",
    "cemilan" => "required",
    "harga" => "required",
    "deskripsi" => "required",
]);

$data = $request->all();
$menus->fill($data);
$menus->save();

```

Selanjutnya merupakan function update atau PUT method. Ini berguna untuk melakukan update terhadap data yang telah di masukan. `$menus = Menu::find $id` berfungsi untuk menemukan data dengan id yang dimaksud. `return response ->json 'message'`

`=> 'Data not found, Please check the ID!' 404` memiliki fungsi untuk mengembalikan response berupa message atau pesan data tidak ditemukan apabila user memasukan data yang belum menjadi bagian dari database. `$this->validate $request` berfungsi untuk melakukan validasi dan data harus di masukan ulang sesuai dengan ketentuan. Kemudian pada line terakhir secara berurutan adalah untuk melakukan pemanggilan data secara keseluruhan untuk kemudian dilakukan pengisian data dan kemudian di simpan kembali kedalam database.

```

public function destroy($id)
{
    $menus = Menu::find($id);           if
(! $menus) {
        return response()->json(['message' => 'Data not found , Please check
the ID'], 404);
    }
    $menus->delete();
    return response()->json(['message' => 'Data has been DELETED'], 200);
}
}

```

Terakhir merupakan function destroy atau DELETE method. Proses menghapus data dilakukan dengan pengambilan id terhadap data yang akan di hapus. Jika data tidak ditemukan maka system akan mengembalikan response berupa message 'Data not found , Please check the ID' . Berikutnya adalah ketika id yang di masukan benar, maka system akan melakukan delete data dalam database 'pet' dan mengembalikan response berupa message 'Data has been DELETED' ketika proses delete berhasil dilakukan.

### Web.php

Setelah membuat function tentunya kita akan membuat endpoint agar method yang telah di deskripsikan dapat dijalankan sesuai dengan fungsinya. Untuk endpoint akan mengikuti banyaknya method atau function yang telah di deskripsikan atau dalam kasus ini terdapat lima endpoint berbeda. Berikut merupakan tampilan source code yang telah kami buat dalam file Web.php.

```

$router->get('/menus', 'MenuController@index');
$router->post('/menus', 'MenuController@store');
$router->get('/menus/{id}', 'MenuController@show');
$router->put('/menus/{id}', 'MenuController@update');
$router->delete('/menus/{id}', 'MenuController@destroy');

```

Penjelasannya kami sajikan kembali dalam tabel seperti yang telah dicantumkan pada bagian sebelumnya ( tabel 1 ).

Endpoint	Description
/menus	Get list of all post
/menus/{id}	Get single post
/menus	Create new post / data
/menus/{id}	Update single post / data
/menus/{id}	Delete single post / data

Table 2

### Menu.php

Dalam Menu.php ini merupakan model yang berguna untuk mewakili pemanggilan data. Jika model sudah selesai dibuat, maka model sudah dapat kita panggil di controller. Untuk memanggil sebuah model di Controller, kita harus mendefinisikan model yang ingin digunakan pada statement use di bagian atas controller, setelah itu barulah model dapat digunakan. Lantas yang perlu diperhatikan dalam penggunaan model adalah, Model harus dibuat mewakili setiap tabel yang kita buat. Model tersebut harus memiliki property \$table, dan \$fillable agar dapat digunakan dengan baik. Ketika model tersebut dipanggil, kita harus mendefinisikannya melalui statement use dengan mengetikkan namespace beserta nama classnya, barulah model dapat dipanggil di controller. Berikut merupakan contoh penerapan model dalam project yang kami kembangkan.

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Menu extends Model
{
    protected $table = 'menus';

    /**
     * The attributes that are mass
     * assignable.
     *
     * @var array
     */
    protected $fillable = array( 'id','kode_makanan',
'kode_minuman','cemilan','harga','deskripsi',
    );
    public $timestamps = true;
}

```

- `protected $table = 'nama_tabel';` berfungsi untuk mendefinisikan model tersebut mewakili tabel apa.
- `protected $fillable = ['kolom_a', 'kolom_b', 'kolom_c', ...];` berfungsi untuk mendefinisikan kolom apa saja yang terdapat pada tabel tersebut.

### User.php

Dalam User.php ini merupakan pengaturan hak akses dari pengguna atau user yang nantinya berguna untuk memproteksi data dari pengguna yang tidak memiliki kepentingan. User harus terdaftar terlebih dahulu agar dapat melakukan proses CRUD pada data. Berikut source codenya.



```

<?php namespace
App\Models;
    use Illuminate\Auth\Authenticatable; use
Illuminate\Contracts\Auth\Access\Authorizable as AuthorizableContract;
use Illuminate\Contracts\Auth\Authenticatable as AuthenticatableContract;
use Illuminate\Database\Eloquent\Factories\HasFactory; use
Illuminate\Database\Eloquent\Model; use Laravel\Lumen\Auth\Authorizable;

class User extends Model implements AuthenticatableContract,
AuthorizableContract
{
    use Authenticatable, Authorizable,
HasFactory;

    /**
     * The attributes that are mass assignable.
     *
     * @var string[]
     */
    protected
$fillable = [
        'name', 'email',
    ];

    /**
     * The attributes excluded from the model's JSON form.
     *
     * @var string[]
     */
    protected
$hidden = [
        'password',
    ];
}

```

## The Result

Berikut merupakan gambaran output ketika melakukan semua Endpoint dan Method yang telah deprogram sebelumnya. Untuk lebih jelasnya dapat dilihat pada penjelasan berikut ini.

### 1. POST Method

Akses untuk melakukan method post adalah melalui link *localhost:8000/menus*. Pertama, user akan memasukan data kedalam body -> raw sesuai dengan ketentuan dan data yang tersedia. Selanjutnya user tinggal melakukan send dan berikut hasilnya.

The image displays two screenshots of a REST client interface, showing successful POST requests.

**Top Screenshot:**

- Method:** POST
- URL:** http://localhost:8000/menus
- Body:** JSON
- Request Body:**

```
1 {
2   "kode_makanan": "A4",
3   "kode_minuman": "B4",
4   "cemilan": "C4",
5   "harga": "80k",
6   "deskripsi": "Paket 1 orang kumplit"
7 }
```
- Status:** 200 OK
- Time:** 169 ms
- Size:** 424 B
- Response Body:**

```
1 {
2   "kode_makanan": "A4",
3   "kode_minuman": "B4",
4   "cemilan": "C4",
5   "harga": "80k",
6   "deskripsi": "Paket 1 orang kumplit",
7   "updated_at": "2023-02-02T22:32:34.000000Z",
8   "created_at": "2023-02-02T22:32:34.000000Z",
9   "id": 4
10 }
```

**Bottom Screenshot:**

- Method:** POST
- URL:** localhost:8000/pet
- Body:** JSON
- Request Body:**

```
1 {
2   "checkup_id": "IMF022023",
3   "pet_name": "Halma ",
4   "pet_age": " 5 Month ",
5   "pet_disease": " Shading Problem ",
6   "pet_gender": " Female ",
7   "owner_name": " Robin ",
8   "doctor_name": " Drh.Imam Fauzi ",
9   "notes": ""
10 }
```
- Status:** 200 OK
- Time:** 189 ms
- Size:** 513 B
- Response Body:**

```
1 {
2   "checkup_id": "IMF022023",
3   "pet_name": " Halma ",
4   "pet_age": " 5 Month ",
5   "pet_disease": " Shading Problem ",
6   "pet_gender": " Female ",
7   "owner_name": " Robin ",
8   "doctor_name": " Drh.Imam Fauzi ",
9   "notes": ""
10 }
```













Image 1 - when data correct



Image 2 - when data incorrect

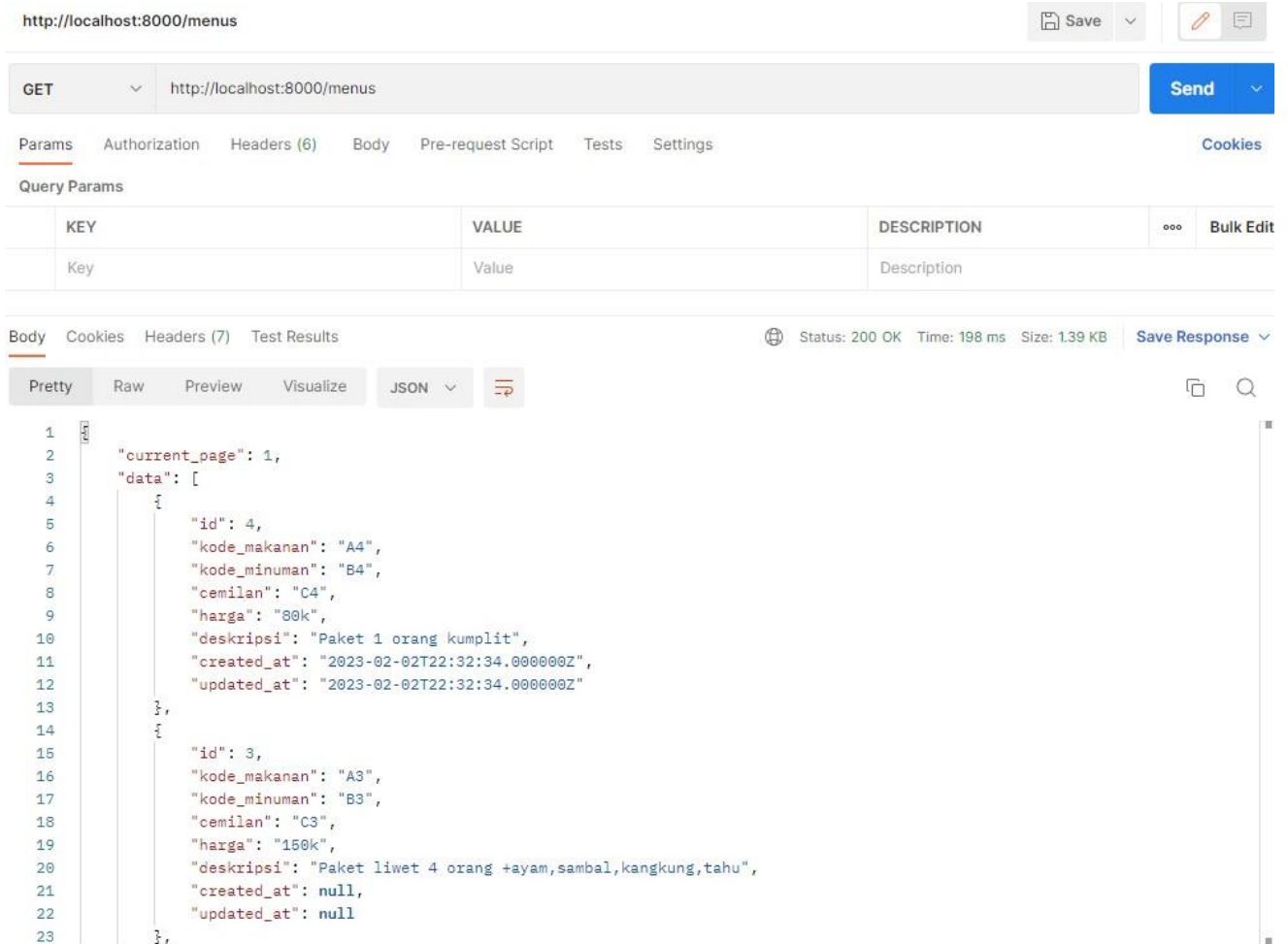
Gambar 1 menunjukkan data ketika berhasil dilakukan POST method. Sedangkan pada gambar 2 merupakan tampilan ketika data tidak berhasil dilakukan POST method. Hal ini terjadi karena checkup\_id telah di atur untuk hanya menerima 1 data saja dan tidak boleh ada data yang sama ( bersifat unique ). Setelah itu kita dapat melakukan check pada database untuk memastikan apakah data benar – benar sudah ter-upload atau belum. Berikut merupakan tampilan database.

+ Opsi

				id	kode_makanan	kode_minuman	cemilan	harga	deskripsi	created_at	updated_at
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	1	A1	B1	C1	100k	Paket lengkap ayam + nasi 2 porsi, es teh manis 2	NULL	NULL
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	2	A2	B2	C2	50k	Paket es mojito 2 + Rujak cireng & bala bala	NULL	NULL
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	3	A3	B3	C3	150k	Paket liwet 4 orang + ayam,sambal,kangkung,tahu	NULL	NULL
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	4	A4	B4	C4	80k	Paket 1 orang kumplit	2023-02-03 05:32:34	2023-02-03 05:32:34

## 2. GET Method ( all )

Akses untuk melakukan method post adalah melalui link *localhost:8000/menus*.



http://localhost:8000/menus

GET http://localhost:8000/menus

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

Query Params

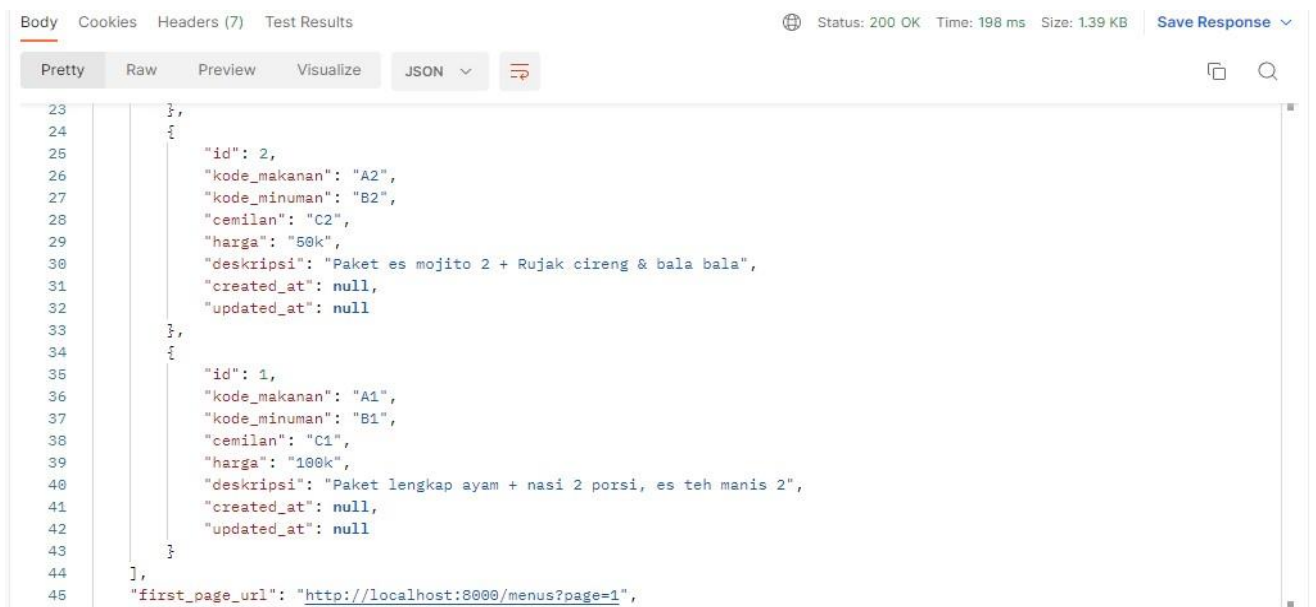
KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 198 ms Size: 1.39 KB Save Response

Pretty Raw Preview Visualize JSON

```
1  {
2    "current_page": 1,
3    "data": [
4      {
5        "id": 4,
6        "kode_makanan": "A4",
7        "kode_minuman": "B4",
8        "cemilan": "C4",
9        "harga": "80k",
10       "deskripsi": "Paket 1 orang kumplit",
11       "created_at": "2023-02-02T22:32:34.000000Z",
12       "updated_at": "2023-02-02T22:32:34.000000Z"
13     },
14     {
15       "id": 3,
16       "kode_makanan": "A3",
17       "kode_minuman": "B3",
18       "cemilan": "C3",
19       "harga": "150k",
20       "deskripsi": "Paket liwet 4 orang +ayam,sambal,kangkung,tahu",
21       "created_at": null,
22       "updated_at": null
23     }
24   ]
25 }
```

Selanjutnya user tinggal melakukan send dan berikut hasilnya.



Body Cookies Headers (7) Test Results Status: 200 OK Time: 198 ms Size: 1.39 KB Save Response

Pretty Raw Preview Visualize JSON

```
23   },
24   {
25     "id": 2,
26     "kode_makanan": "A2",
27     "kode_minuman": "B2",
28     "cemilan": "C2",
29     "harga": "50k",
30     "deskripsi": "Paket es mojito 2 + Rujak cireng & bala bala",
31     "created_at": null,
32     "updated_at": null
33   },
34   {
35     "id": 1,
36     "kode_makanan": "A1",
37     "kode_minuman": "B1",
38     "cemilan": "C1",
39     "harga": "100k",
40     "deskripsi": "Paket lengkap ayam + nasi 2 porsi, es teh manis 2",
41     "created_at": null,
42     "updated_at": null
43   }
44 ],
45 "first_page_url": "http://localhost:8000/menus?page=1",
```

Image 4 - Get all

Dalam proses tersebut dapat dikatakan telah berhasil, karena semua data di dalam database ( termasuk yang baru saja ditambahkan ) telah muncul pada bagian response.

### 3. Get Method ( by Id )

Akses untuk melakukan method post adalah melalui link *localhost:8000/menus/{id}*.

Selanjutnya user tinggal melakukan send dan berikut hasilnya.

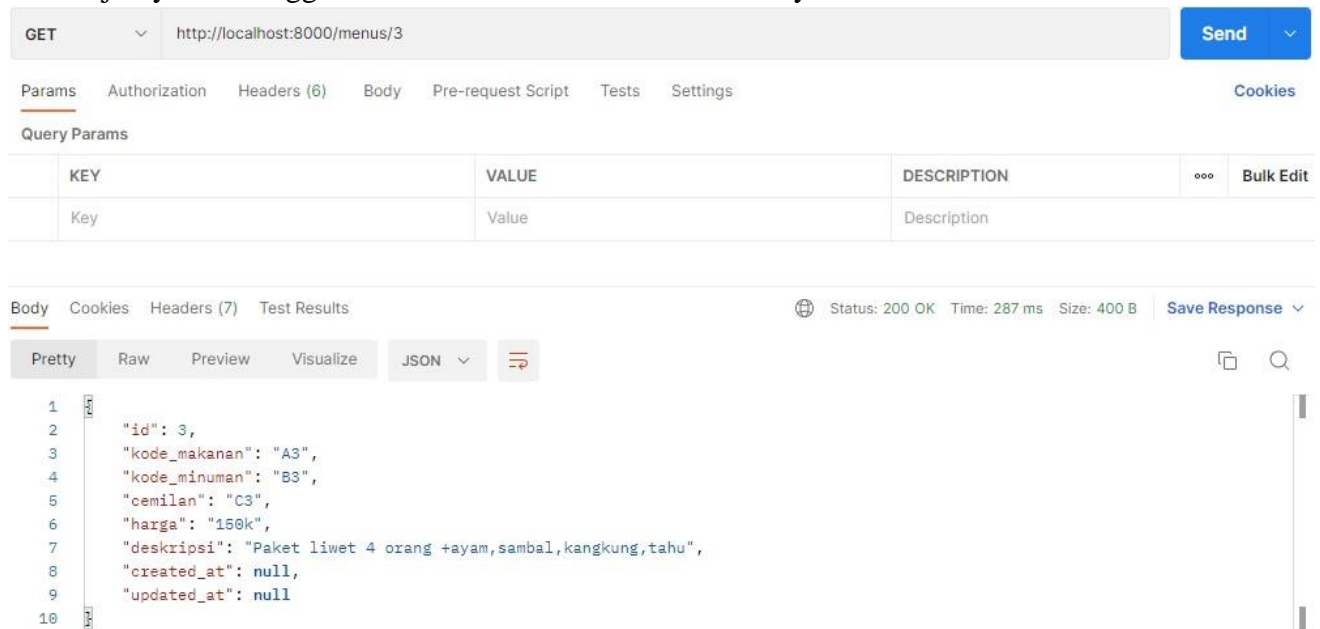


Image 5 - when id is correct

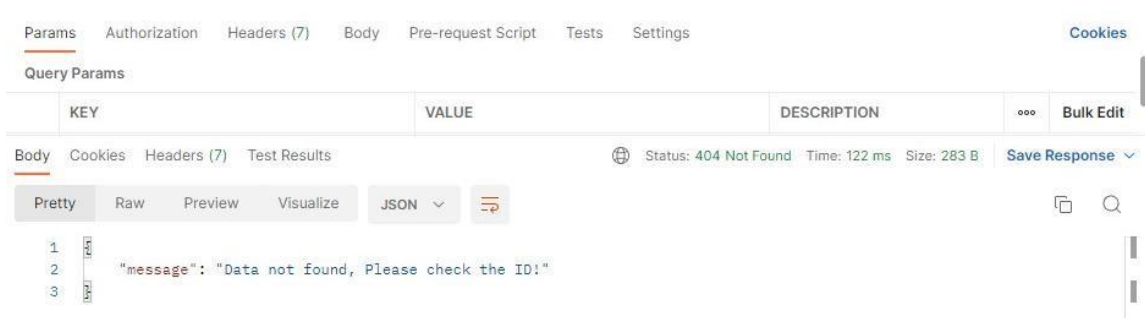
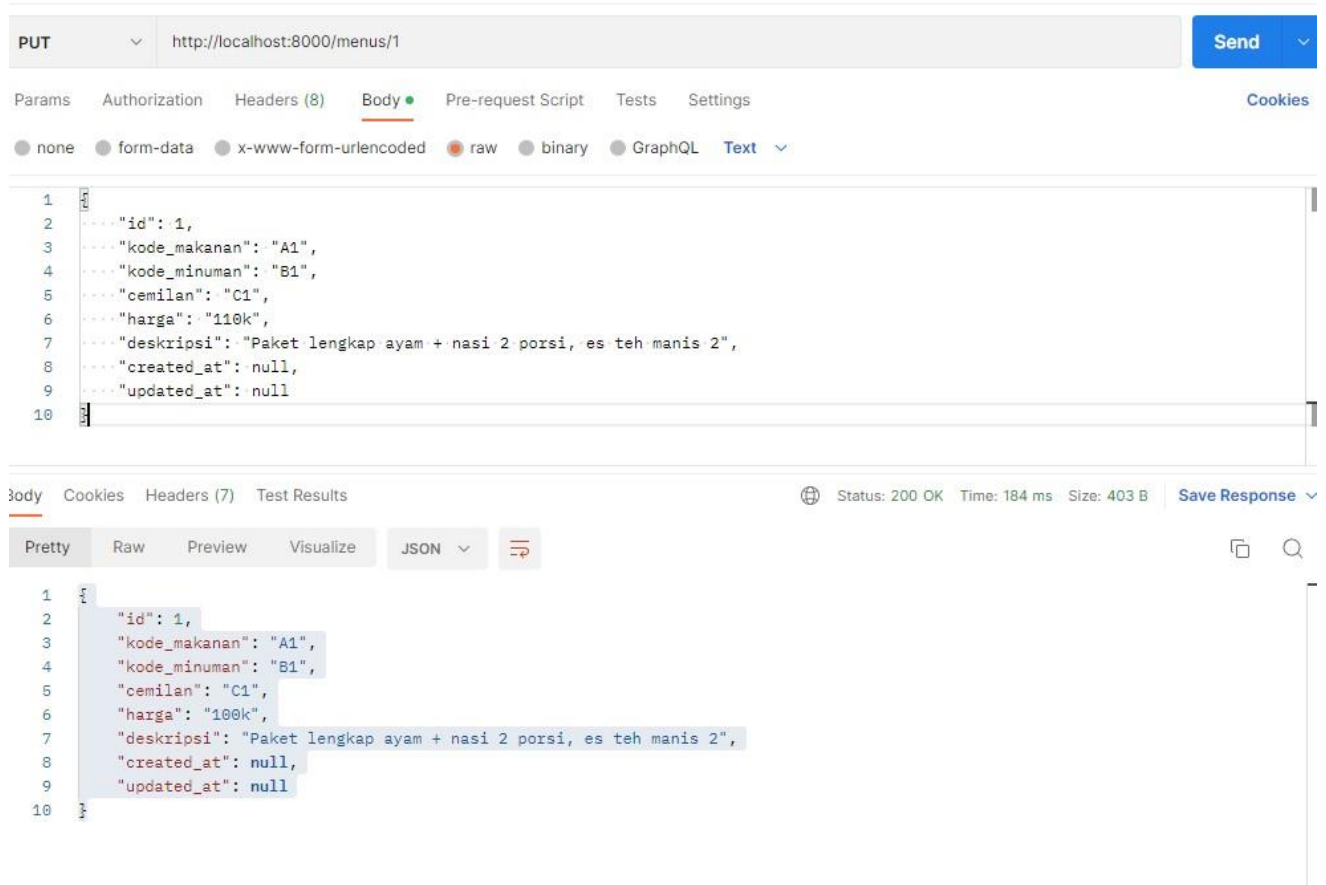


Image 6 - when id is incorrect

Gambar 5 menunjukan data ketika berhasil dilakukan GET method. Sedangkan pada gambar 6 merupakan tampilan ketika data tidak berhasil dilakukan GET method. System akan mengirimkan return response berupa message. Hal ini terjadi karena id akan berbeda secara default setiap datanya, maka id tidak akan menjadi sama antar data.

#### 4. PUT Metod

PUT method atau biasa dikenal dengan method Update merupakan method yang digunakan untuk melakukan update terhadap data yang telah dimasukkan sebelumnya. Jika data terjadi kesalahan, maka method ini dapat menjadi solusi agar tidak perlu menghapus data sebelumnya. Dalam case ini user hanya tinggal memasukkan id dari data yang akan dilakukan proses update di akhir link dan memasukkan ulang data yang diperlukan. Untuk lebih jelasnya, berikut kami sajikan contohnya dalam bentuk screenshot hasil percobaan.



Pada gambar 7 tersebut, data telah berhasil dilakukan update. Update yang dilakukan adalah pada jenis penyakit dari hewan. Dapat kita pastikan pula dalam database apakah sudah terupdate atau belum.

+ Opsi

				id	kode_makanan	kode_minuman	cemilan	harga	deskripsi	created_at	updated_at
<input type="checkbox"/>	Ubah	Salin	Hapus	1	A1	B1	C1	110k	Paket lengkap ayam + nasi 2 porsi, es teh manis 2	NULL	2023-02-03 05:45:30
<input type="checkbox"/>	Ubah	Salin	Hapus	2	A2	B2	C2	50k	Paket es mojito 2 + Rujak cireng & bala bala	NULL	NULL
<input type="checkbox"/>	Ubah	Salin	Hapus	3	A3	B3	C3	150k	Paket liwet 4 orang + ayam, sambal, kangkung, tahu	NULL	NULL
<input type="checkbox"/>	Ubah	Salin	Hapus	4	A4	B4	C4	80k	Paket 1 orang kemplit	2023-02-03 05:32:34	2023-02-03 05:32:34

☐ Pilih Semua
 Dengan pilihan:
 ☐ Ubah
 ☐ Salin
 ☐ Hapus
 ☐ Ekspor

Saat di lakukan pengecekan pada database server, data sudah berhasil ter-update. Dapat di perhatikan pula pada gambar 7 dan 8 jenis penyakit sudah sama yang menandakan data berhasil dilakukan update. Dibawah ini merupakan gambar ketika user memasukan data yang tepat, tetapi dengan id yang tidak tepat. System akan memberikan return response berupa message seperti berikut ini

PUT

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

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

```

1  {
2    "id": 1,
3    "kode_makanan": "A1",
4    "kode_minuman": "B1",
5    "cemilan": "C1",
6    "harga": "110k",
7    "deskripsi": "Paket lengkap ayam + nasi 2 porsi, es teh manis 2",
8    "created_at": null,
9    "updated_at": null
10 }
  
```

Body Cookies Headers (7) Test Results

☒ Pretty
 ☐ Raw
 ☐ Preview
 ☐ Visualize
 ☐ JSON

```

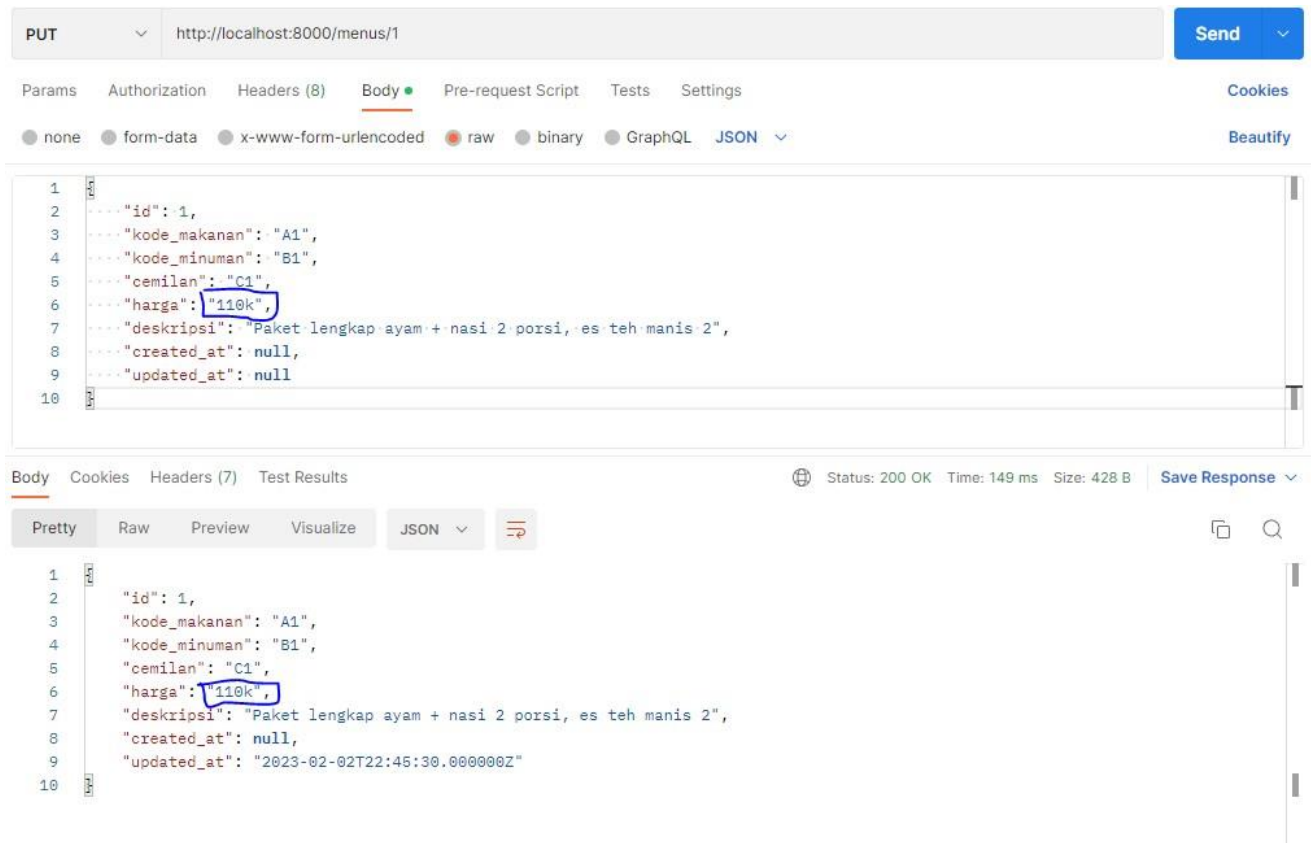
1  {
2    "id": 1,
3    "kode_makanan": "A1",
4    "kode_minuman": "B1",
5    "cemilan": "C1",
6    "harga": "110k",
7    "deskripsi": "Paket lengkap ayam + nasi 2 porsi, es teh manis 2",
8    "created_at": null,
9    "updated_at": "2023-02-02T22:45:30.000000Z"
10 }
  
```

Image 9 - incorrect id

Selanjutnya merupakan gambaran ketika user tidak mengisi field dengan benar ( kosong ). System akan memberikan return response berupa message seperti pada gambar 10. Ini terjadi



karena dalam MenuController.php telah di deklarasikan bahwa setiap field harus diisi dengan baik atau tidak dapat di kosongkan sebagai syarat ( require ) untuk melakukan PUT method.



## 5. DELETE Method

Kita telah memasuki method terakhir, yaitu DELETE atau biasa dikenal dengan Destroy. Method ini memungkinkan user untuk melakukan penghapusan data yang sudah tidak diperlukan lagi. Melalui link seperti pada gambar 11, serta menyertakan id terhadap data yang akan dihapus, maka user akan dapat menghapus data tersebut. Untuk lebih jelasnya dapat dilihat contoh penggunaannya pada gambar berikut.



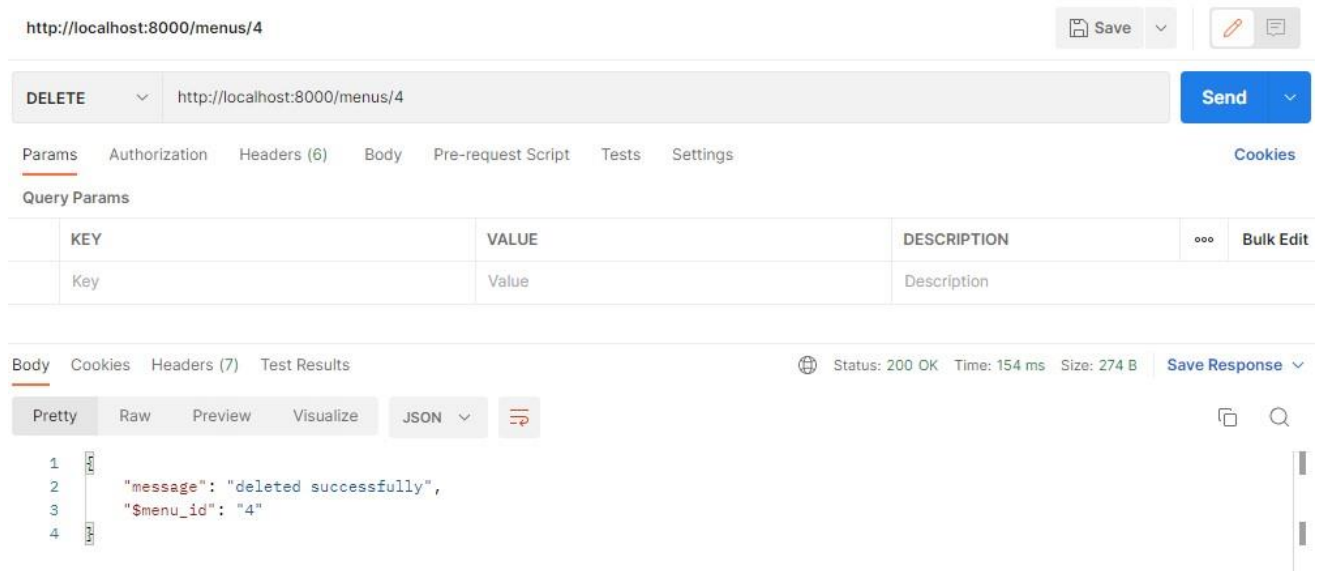


Image 11 - data deleted

Jika id yang dimasukan telah sesuai, maka tampilannya akan seperti gambar 11. Data berhasil terhapus dan muncul message sebagai return response. Sebaliknya, jika id yang dimasukan tidak tepat maka akan muncul response seperti pada gambar 12 dibawah ini.

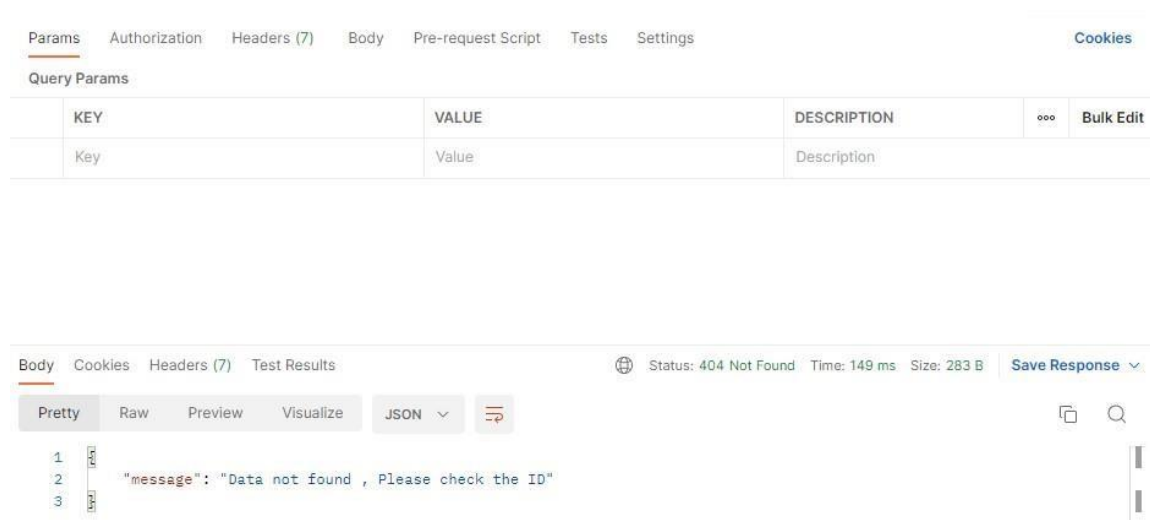
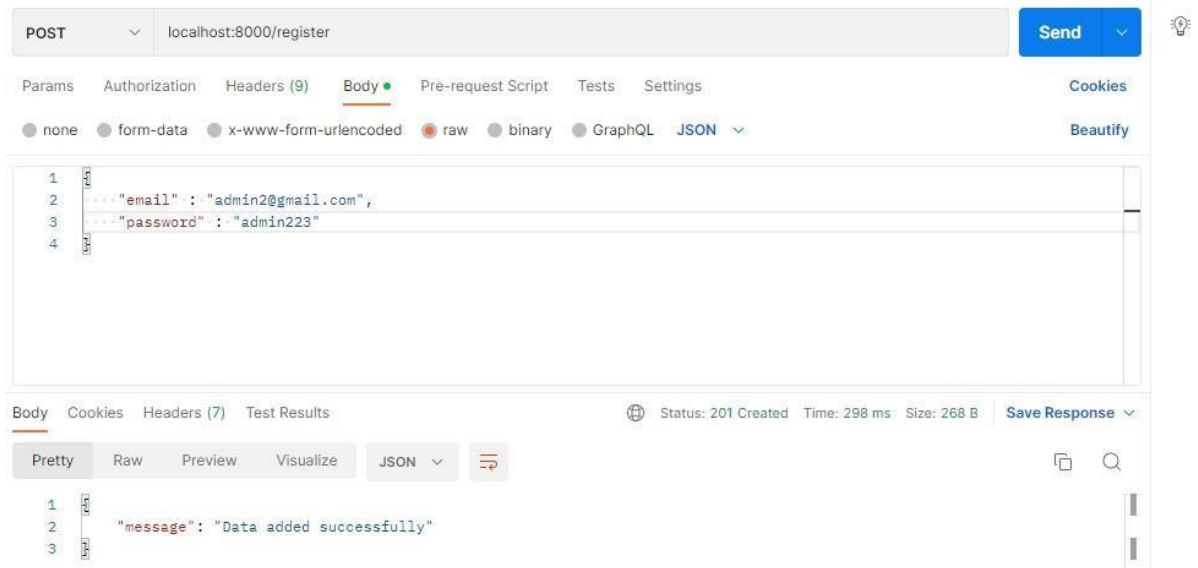


Image 12 - incorrect id

## 6. User Authentication

User Authentication atau biasa dikenal dengan Login form merupakan langkah untuk melindungi data dari user atau pengguna yang tidak diinginkan. Dalam user authentication terdapat email dan password yang harus terdaftar pada database agar nantinya user dapat melakukan proses CRUD pada data. System akan memberikan token yang dapat digunakan oleh user yang telah terdaftar untuk melakukan proses login yang nantinya menuju kepada proses edit data. Sebagai gambaran, berikut merupakan hasil yang telah kami coba dengan beberapa perbandingan yang akan dijelaskan kemudian.

1. User akan diminta untuk melakukan register atau registrasi dengan cara memasukkan email dan password pada halaman register seperti pada gambar berikut ini.



Gambar 13

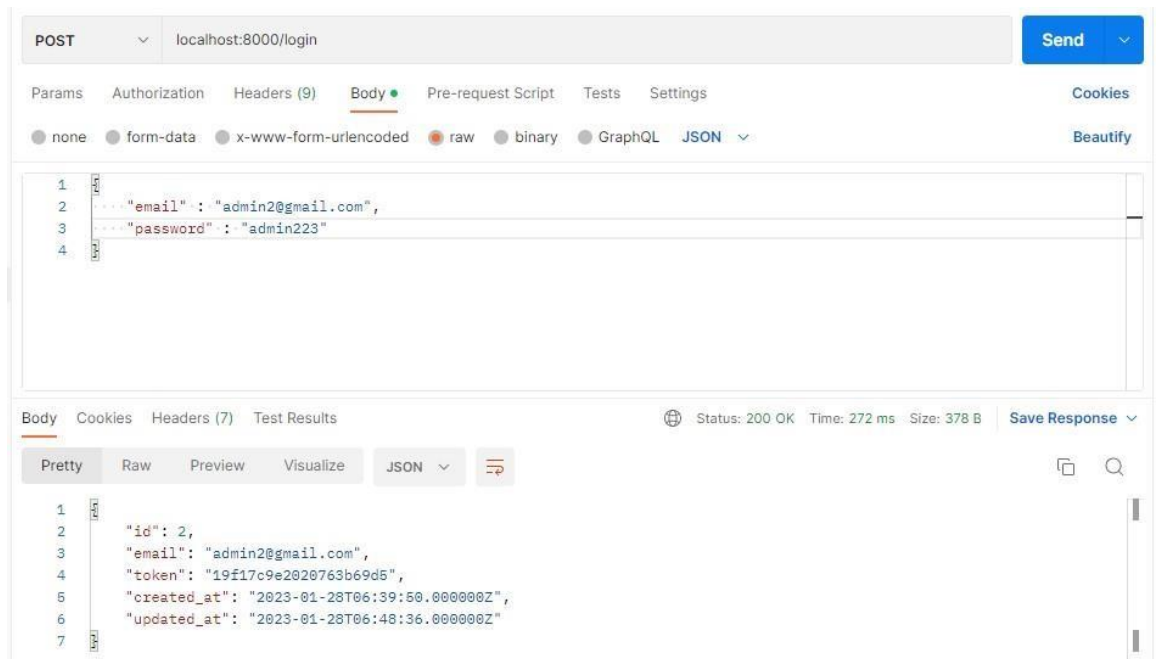
Dengan method POST, user hanya tinggal mengirimkan data kepada database. Setelah dilakukan send atau pengiriman data berhasil, maka akan system akan memberikan return response berupa message seperti pada gambar di atas. Dan pada gambar dibawah ini merupakan hasil pada database.

	id	email	password	token	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	admin1@gmail.com	\$2y\$10\$npF30i8UDdwGik3qKO4pYOn6bUm8jFpbqzdVfFURYfe...	07de4d602a6970a3c881	2023-01-28 04:37:53	2023-01-28 05:03:29
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	admin2@gmail.com	\$2y\$10\$QTSunqKg7MZRUjnjE1fcQ.Jcljq89ad/X2axSvmArSV...	NULL	2023-01-28 13:39:50	2023-01-28 13:39:50

Gambar 14

Data telah berhasil masuk kedalam database, dapat diperhatikan admin 2 telah berhasil ditambahkan.

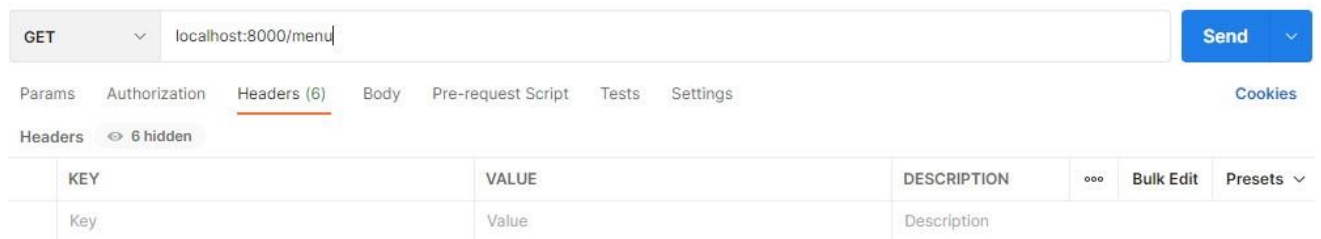
2. Selanjutnya user memerlukan token untuk dapat login ke system dan melakukan CRUD. Untuk mendapatkan token yang di generate secara otomatis oleh system, maka user dapat melakukan POST dengan memasukkan email dan password yang telah terdaftar seperti pada langkah sebelumnya. Untuk lebih jelas dapat dilihat pada gambar berikut ini.



Gambar 15

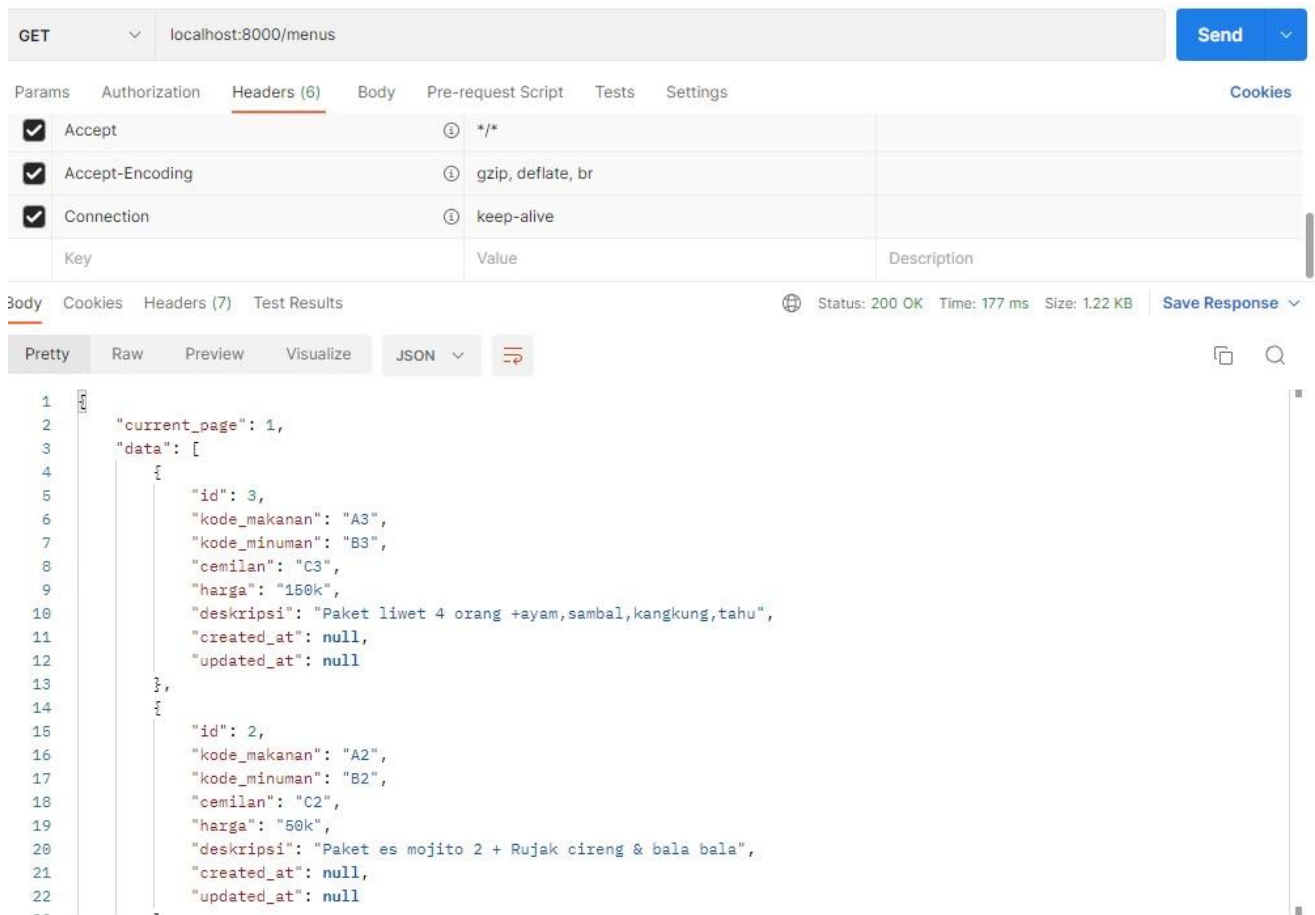
Dengan otomatis ketika user telah melakukan POST maka system akan memberikan token yang dibuat secara random.

3. Pada proses berikutnya, user tinggal memasukkan token yang telah diberikan system pada kolom Headers ketika ingin melakukan proses CRUD seperti pada gambar berikut.

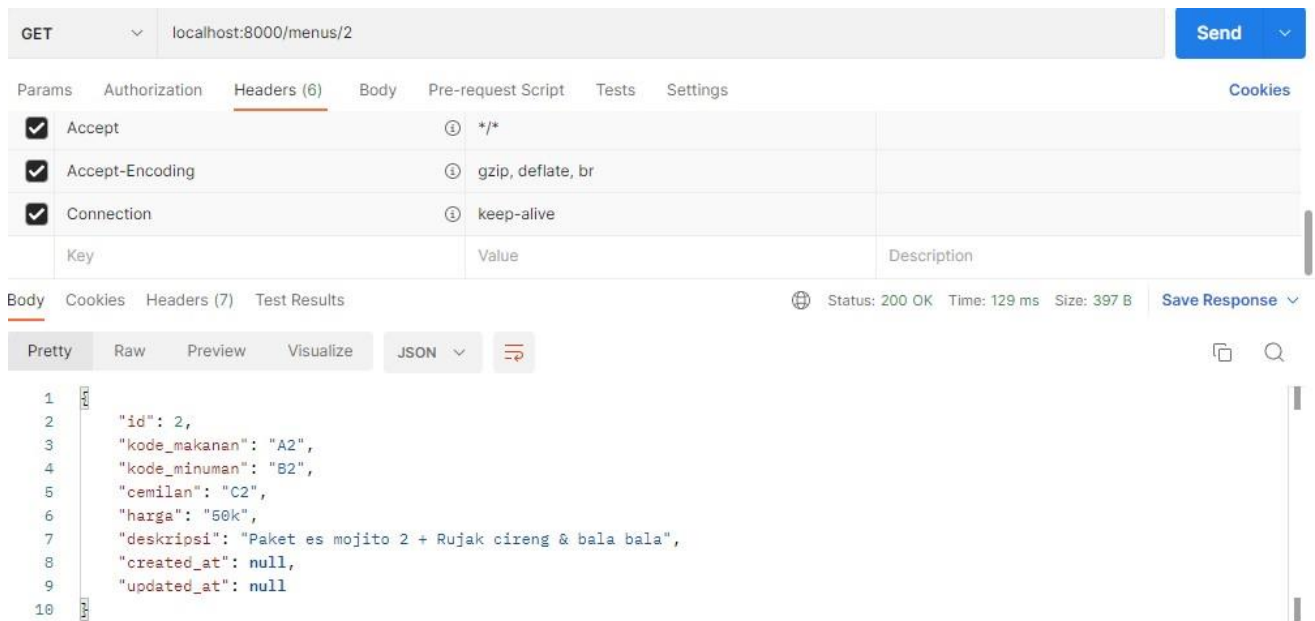


Gambar 16

Gambar dibawah ini merupakan proses ketika dilakukan GET all data sebagai contoh penerapannya.

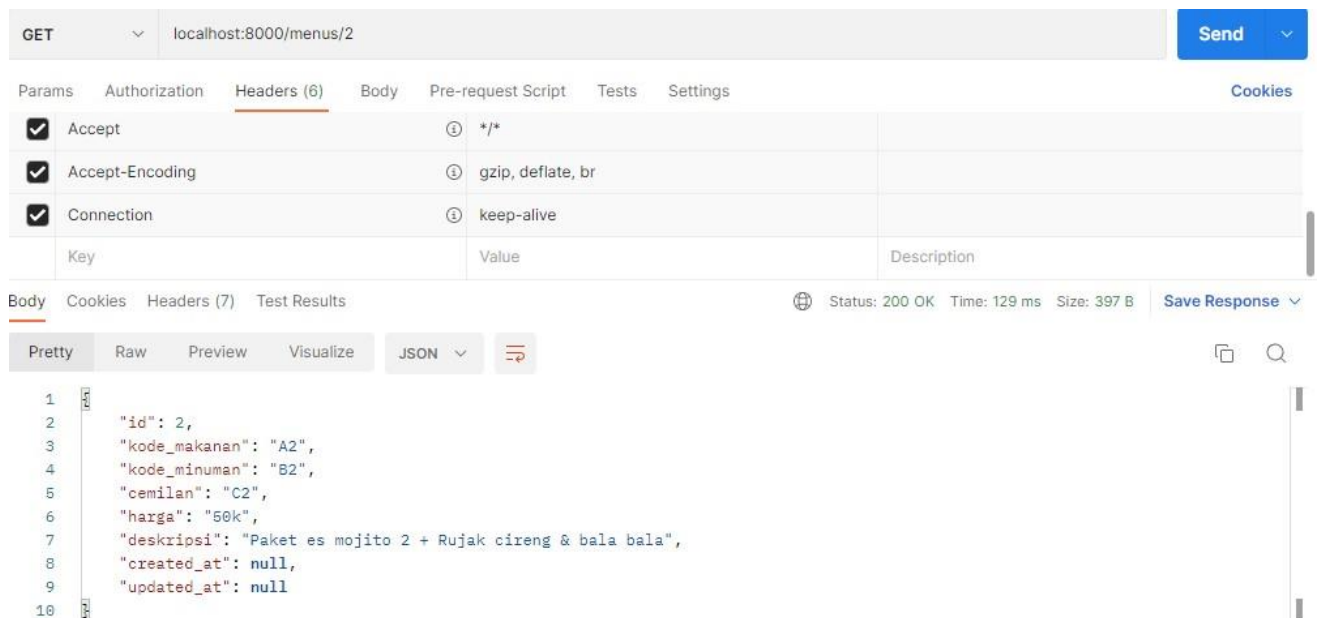


Gambar 17 – Get all data – success

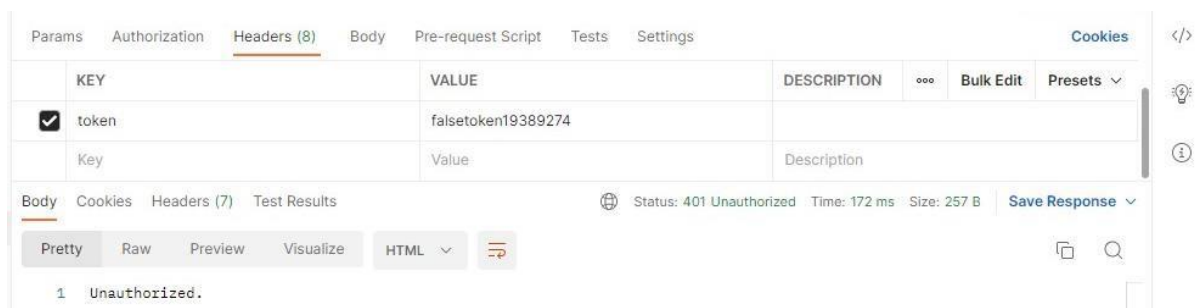


Gambar 18 – Get data by Id - success

Dan pada gambar berikut merupakan gambaran yang terjadi jika token yang dimasukan tidak sesuai dengan apa yang telah diberikan oleh system. System akan memberikan return response seperti yang tertera pada gambar.



Gambar 19 – Get all data - unauthorized



Gambar 20 - Get data by ID – unauthorized

Itulah hasil dari endpoint yang kami kembangkan terhadap proses CRUD dari project *CRUD-in aja*. Semoga dapat dimengerti dan dapat menjadi acuan dalam pengembangan proses lainnya.

## CHAPTER FIVE

### RANCANGAN ERD

#### Get to Know About an ERD ?

ERD atau *Entity Relationship Diagram* merupakan suatu diagram yang menunjukkan informasi yang dibuat, disimpan, dan digunakan dalam sistem bisnis atau sistem lainnya untuk memperlihatkan hubungan atau relasi antar entitas atau objek yang terlihat beserta atributnya. Sederhananya, entity relationship diagram adalah salah satu jenis diagram yang sifatnya lebih struktural dan bisa digunakan untuk dimanfaatkan dalam suatu desain pada suatu database ataupun pada sebuah business plan.

Selain itu, terdapat banyak sekali komponen yang menjadikan entity relationship diagram, seperti konektor dan juga simbol yang berbeda-beda. Nantinya, komponen ini akan melakukan visualisasi pada dua informasi yang dianggap penting. Informasi pertama adalah entitas utama yang terdapat di dalam ruang lingkup suatu sistem. Sedangkan untuk informasi kedua adalah hubungan yang terdapat di berbagai entitas itu sendiri. ERD juga memiliki beberapa fungsi utama, yaitu :

1. Memberikan kemudahan dalam menganalisis sebuah basis data(database) dengan cara yang cepat dan relatif murah.
2. Menjalankan hubungan antar data yang memiliki keterkaitan berdasarkan objek yang dihubungkan dengan satu relasi.
3. Mendokumentasikan data yang ada dalam sebuah basis data dengan cara menganalisis serta mengidentifikasi setiap objek atau entitas dan relasinya.
4. Melakukan pengujian model yang telah dibuat.

Setelah mengenal fungsi dari ERD, pada bahasan berikut akan membahas tentang komponen susunan dari ERD. Berikut ini merupakan beberapa susunannya.

#### 1. Entitas

Entitas adalah sekumpulan objek yang nantinya akan diidentifikasi. Ketika sedang membuat ERD, umumnya suatu entitas akan digambarkan dalam suatu simbol persegi panjang. Disisi lain, entitas yang lemah akan digambarkan dengan simbol persegi panjang yang kecil di dalam persegi panjang yang lebar.

Masing-masing entitas sudah tentu mempunyai perbedaan. Bila ternyata ada kesamaan, maka entitas tersebut tidak perlu dicantumkan.

#### 2. Atribut

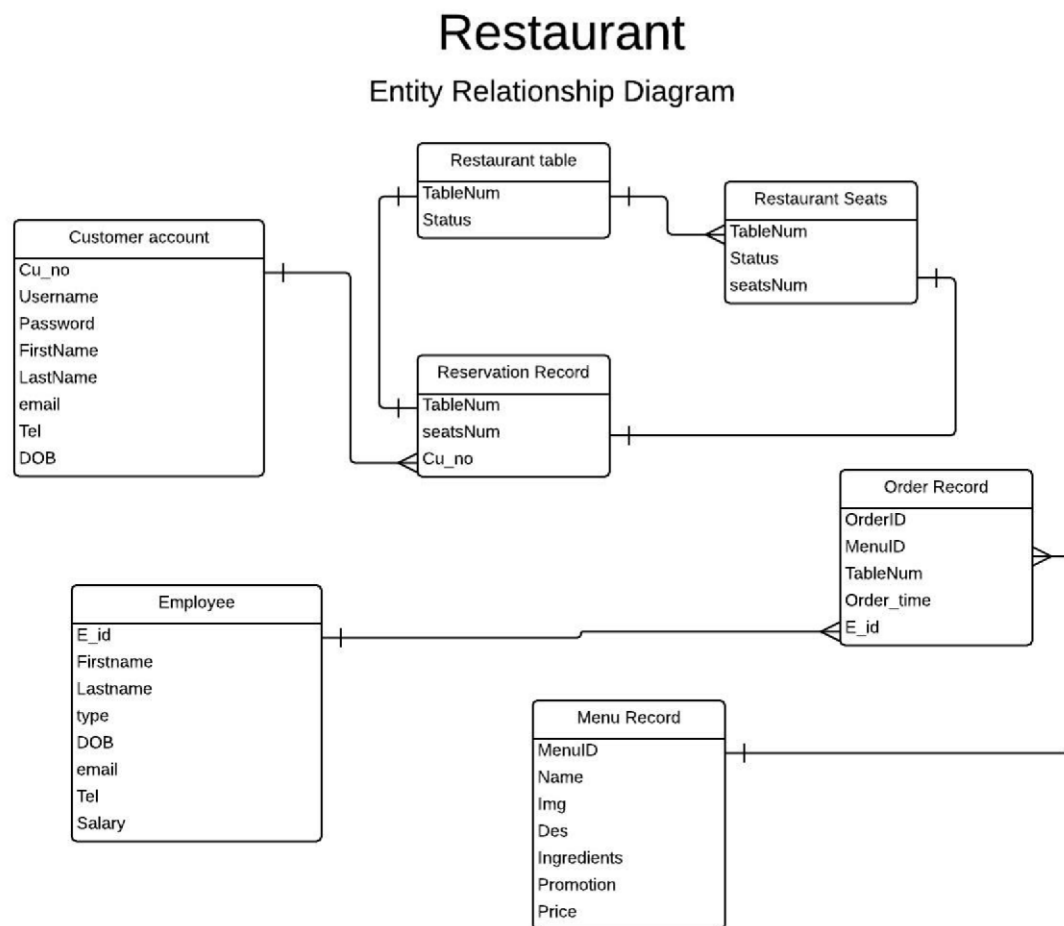
Setiap entitas selalu mempunyai elemen ataupun atribut agar bisa menjelaskannya dengan baik. Umumnya, atribut akan digambarkan sebagai simbol. Di dalam entity relationship diagram, terdapat beberapa jenis atribut, seperti atribut simple, atribut kunci, atribut multivalued, atribut gabungan, dan juga atribut derivatif.

### 3. Relasi

Relasi ataupun hubungan adalah suatu tingkat ketertarikan pada beberapa entitas dari gabungan lainnya. umumnya, relasi ini akan dicerminkan dalam simbol berbentuk belah ketupat. Di dalam entity relationship diagram, relasi ini akan dibagi menjadi beberapa jenis, seperti One to One, One to Many, dan juga Many to Many.

### ERD of Our Project

Berikut ini kami tampilkan gambaran dari ERD project yang tela kami kembangan.



Berikut merupakan penjelasan terhadap ERD Diagram yang telah dibuat.

1. Terdapat empat entitas utama yaitu, Customer, Menu, Cassier, dan Checkup.
2. Didalam setiap entitas memiliki masing masing atribut.

Untuk Customer melihat atribut kode\_makanan dan juga deskripsi.

Untuk Menu memiliki atribut id, kode\_makanan , kode\_minuman, deskripsi, dan harga.

Untuk Cassier memiliki atribut berupa id .

dan Harga yang memiliki atribut.

3. Selanjutnya yaitu setiap entitas memiliki relasi one to many dan many to many yang dapat dilihat pada gambar Erd 1. Untuk penjelasannya sebagai berikut :
  - Customer memiliki relasi one to many.
  - Menu relasi many to many.
  - Cassier memiliki relasi one to many.
  - Dan harga memiliki relasi one to many.



## **CHAPTER SIX**

### **KESIMPULAN**

Dengan menggunakan REST API pada web service kami tujuannya untuk menjadikan sistem memiliki performa yang baik, cepat dan mudah untuk di kembangkan (scale) terutama dalam pertukaran dan komunikasi data.

Permintaan ke API bisa menggunakan GET untuk mengambil sumber daya, PUT untuk mengubah status atau memperbarui sumber daya, yang dapat berupa objek, file, atau blok, POST untuk membuat sumber daya itu, dan DELETE untuk menghapusnya. Penamaan dan struktur URL yang konsisten akan menghasilkan API yang baik dan mudah untuk dimengerti developer.

Beberapa fitur yang dibutuhkan diantaranya adalah fitur input data (makanan, customer dan cassier), fitur update data (makanan, customer dan cassier) fitur hapus data (makanan, customer dan cassier), dan juga fitur untuk menampilkan data yang telah dimasukan sebelumnya baik secara satu per satu maupun tampilan secara keseluruhan data.