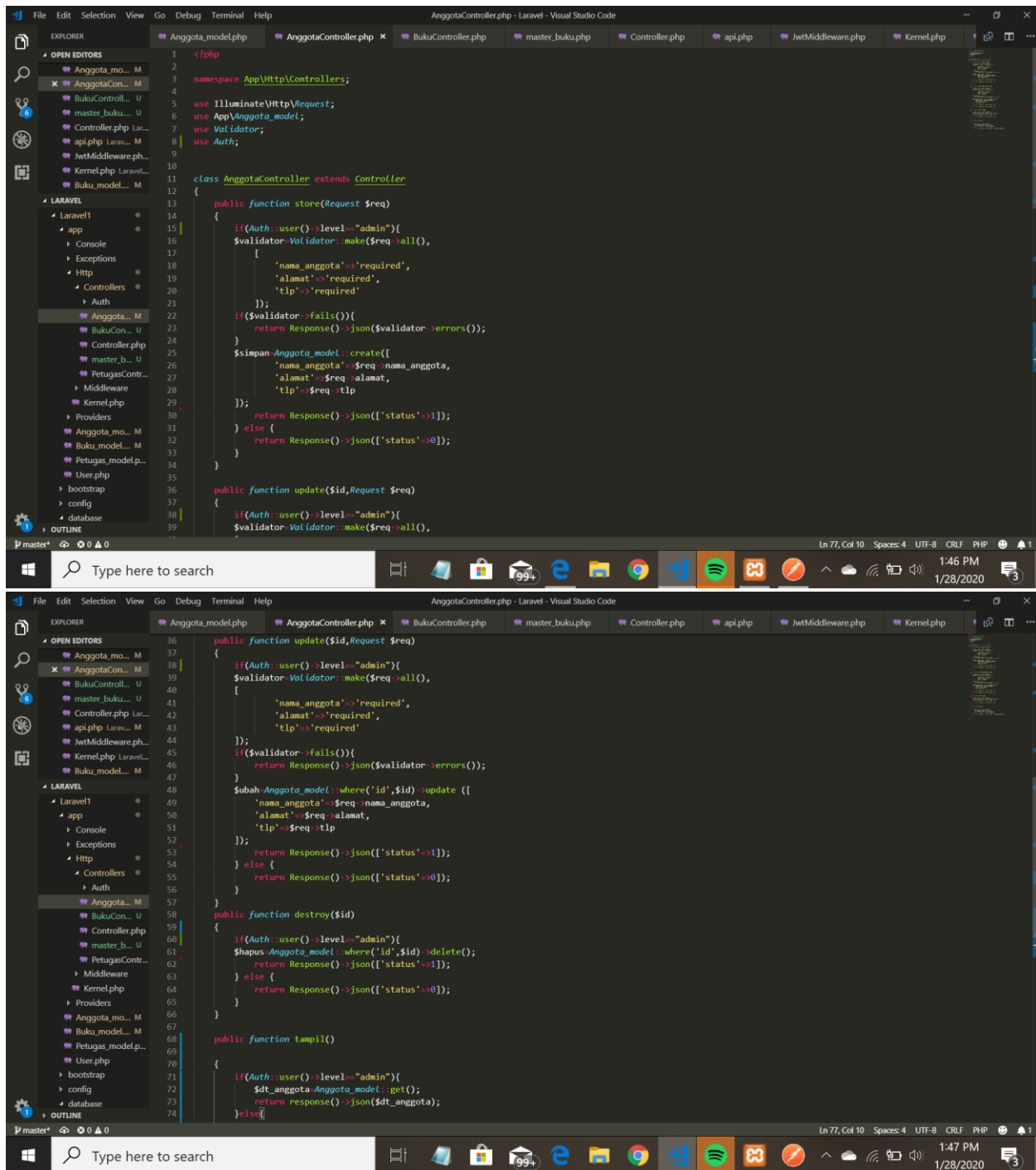


## Controller Anggota



```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Anggota_model;
7 use Validator;
8 use Auth;
9
10
11 class AnggotaController extends Controller
12 {
13     public function store(Request $req)
14     {
15         if(Auth::user()->level=="admin"){
16             $validator=Validator::make($req->all(),
17             [
18                 'nama_anggota'=>'required',
19                 'alamat'=>'required',
20                 'telp'=>'required'
21             ]);
22             if($validator->fails()){
23                 return Response()->json($validator->errors());
24             }
25             $simpan=Anggota_model::create([
26                 'nama_anggota'=>$req->nama_anggota,
27                 'alamat'=>$req->alamat,
28                 'telp'=>$req->telp
29             ]);
30             return Response()->json(['status'=>1]);
31         } else {
32             return Response()->json(['status'=>0]);
33         }
34     }
35
36     public function update($id,Request $req)
37     {
38         if(Auth::user()->level=="admin"){
39             $validator=Validator::make($req->all(),
40             [
41                 'nama_anggota'=>'required',
42                 'alamat'=>'required',
43                 'telp'=>'required'
44             ]);
45             if($validator->fails()){
46                 return Response()->json($validator->errors());
47             }
48             $ubah=Anggota_model::where('id',$id)->update([
49                 'nama_anggota'=>$req->nama_anggota,
50                 'alamat'=>$req->alamat,
51                 'telp'=>$req->telp
52             ]);
53             return Response()->json(['status'=>1]);
54         } else {
55             return Response()->json(['status'=>0]);
56         }
57     }
58
59     public function destroy($id)
60     {
61         if(Auth::user()->level=="admin"){
62             $hapus=Anggota_model::where('id',$id)->delete();
63             return Response()->json(['status'=>1]);
64         } else {
65             return Response()->json(['status'=>0]);
66         }
67     }
68
69     public function tampil()
70     {
71         if(Auth::user()->level=="admin"){
72             $dt_anggota=Anggota_model::get();
73             return response()->json($dt_anggota);
74         } else {
75             return response()->json(['status'=>0]);
76         }
77     }
78 }
```

```
AnggotaController.php - Laravel - Visual Studio Code

EXPLORER
- OPEN EDITORS
  - Anggota_mo... M 41
  - AnggotaCon... M 42
  - BukuControll... U 44
  - master_buku... U 45
  - Controller.php Lar... 46
  - api.php Larav... M 47
  - JwtMiddleware.ph... 48
  - Kernel.php Laravel... 49
  - Buku_model... M 50
  - LARAVEL
    - Laravel1
      - app
        - Console
        - Exceptions
        - HTTP
          - Controllers
            - Auth
              - Anggota... M 51
              - BukuCon... U 52
              - Controller.php 53
              - master_b... U 54
              - PetugasContr... 55
            - Middleware
            - Kernel.php 56
            - Providers
              - Anggota_mo... M 57
              - Buku_model... M 58
              - Petugas_modelp... 59
              - User.php 60
            - bootstrap
            - config
            - database
          - OUTLINE 61
  - master* 62

AnggotaController.php
41 'nama_anggota'=>'required',
42 'alamat'=>'required',
43 'tlp'=>'required'
44 );
45 if($validator->fails()){
46     return Response()->json($validator->errors());
47 }
48 $ubah=Anggota_model::where('id',$id)->update([
49     'nama_anggota'=>$req->nama_anggota,
50     'alamat'=>$req->alamat,
51     'tlp'=>$req->tlp
52 ]);
53 return Response()->json(['status'=>1]);
54 } else {
55     return Response()->json(['status'=>0]);
56 }
57 }
58 public function destroy($id)
59 {
60     if(Auth::user()->level=="admin"){
61         $hapus=Anggota_model::where('id',$id)->delete();
62         return Response()->json(['status'=>1]);
63     } else {
64         return Response()->json(['status'=>0]);
65     }
66 }
67 }
68 public function tampil()
69 {
70     if(Auth::user()->level=="admin"){
71         $idt=Anggota_model::get();
72         return response()->json($idt_anggota);
73     }else{
74         return response()->json(['status'=>'anda bukan admin']);
75     }
76 }
77 }
78 }
79 }
```

## Create Buku

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections APIs BETA

Save Responses Clear all

Today

- POST http://127.0.0.1:8000/api/anggota
- DEL http://127.0.0.1:8000/api/hapus\_buku/7
- GET http://127.0.0.1:8000/api/tampil\_buku
- PUT http://127.0.0.1:8000/api/buku/7
- POST http://127.0.0.1:8000/api/buku
- POST http://127.0.0.1:8000/api/buku
- http://127.0.0.1:8000/api/hapus\_buku/7

POST http://127.0.0.1:8000/api/anggota

Send Save

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> nama_anggota	Aku aja	
<input checked="" type="checkbox"/> alamat	Malang	
<input checked="" type="checkbox"/> tlp	09876542323	
Key	Text	Value
		Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 312ms Size: 289 B Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   "status": 1
3 }
```

## Update Anggota

The screenshot shows the Postman application with a PUT request to `http://127.0.0.1:8000/api/anggota/1`. The request body is in the `x-www-form-urlencoded` format with the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> nama_anggota	aki aki	
<input checked="" type="checkbox"/> alamat	Malang	
<input checked="" type="checkbox"/> tlp	098765458	
Key	Value	Description

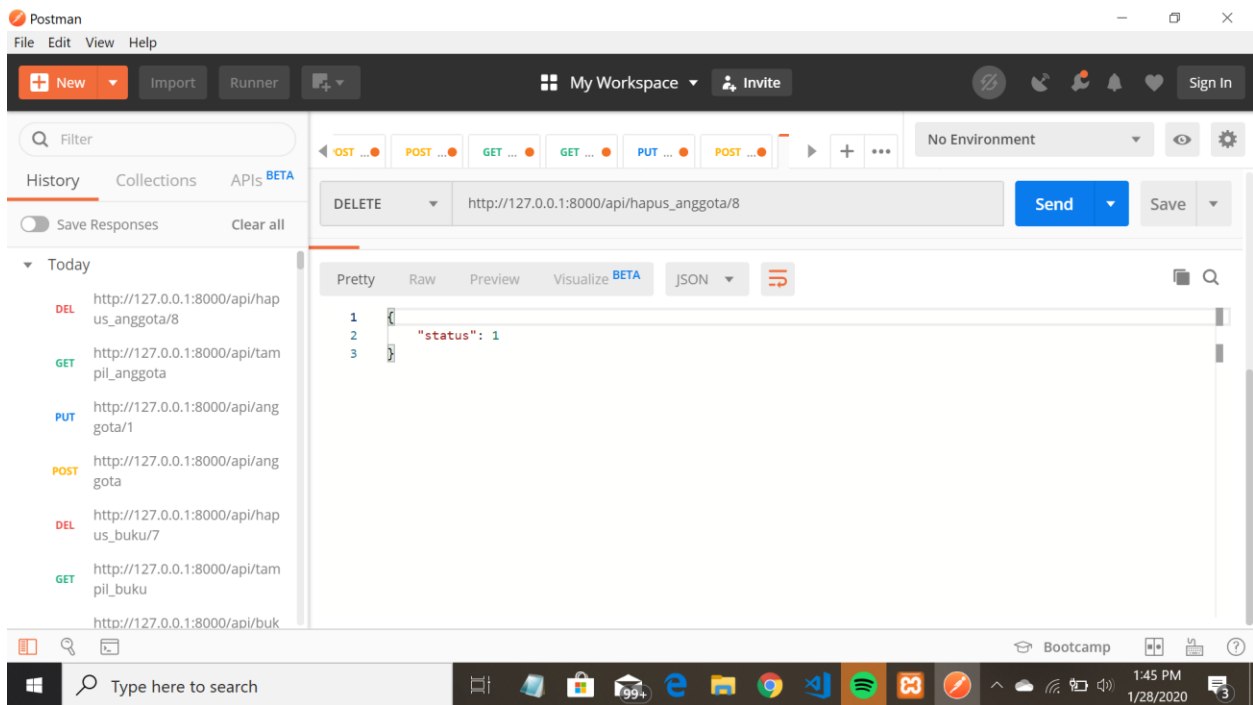
The response is a JSON object: `{ "status": 1 }`. The status is 200 OK, time is 302ms, and size is 289 B.

## Tampil Anggota

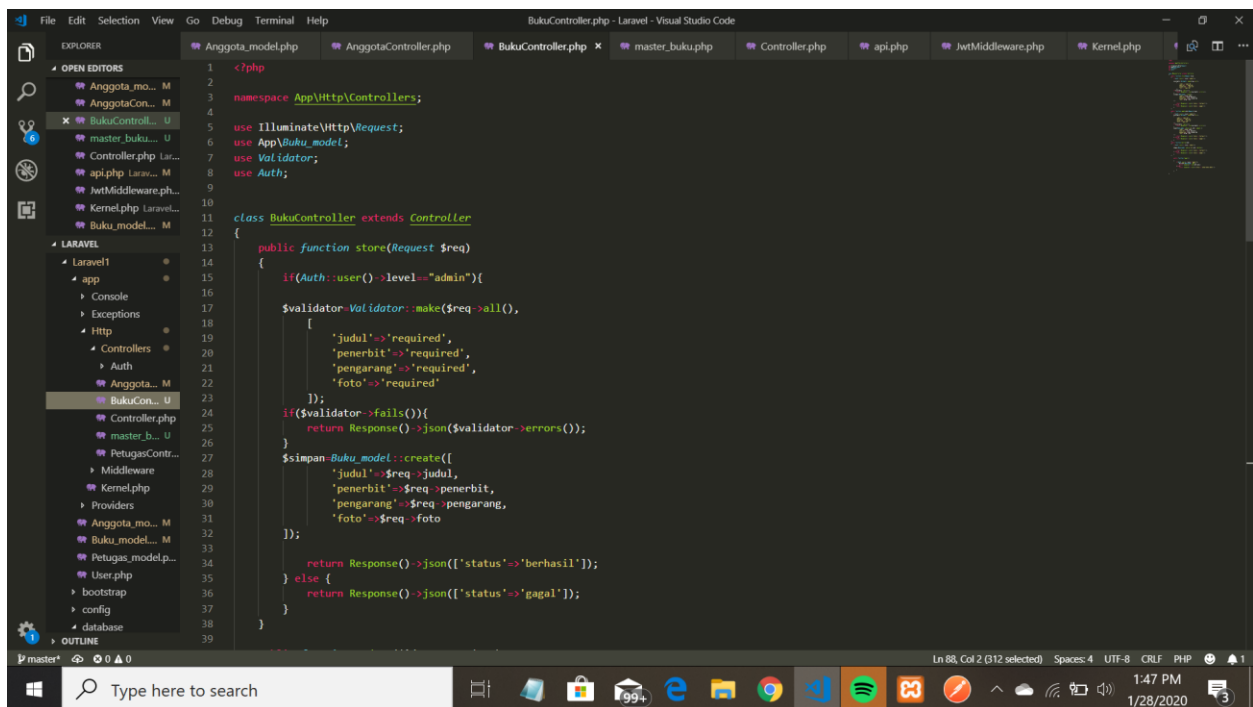
The screenshot shows the Postman application with a GET request to `http://127.0.0.1:8000/api/tampil_anggota`. The response is a JSON array of two member objects:

```
34 {  
35   "id": 7,  
36   "nama_anggota": "Syifa",  
37   "alamat": "Malang",  
38   "tlp": "0891746714065",  
39   "created_at": null,  
40   "updated_at": null  
41 },  
42 {  
43   "id": 8,  
44   "nama_anggota": "Aku aja",  
45   "alamat": "Malang",  
46   "tlp": "09876542323",  
47   "created_at": "2020-01-28 06:42:55",  
48   "updated_at": "2020-01-28 06:42:55"  
49 }  
50
```

## Hapus Anggota



## Controller Buku



BukuController.php - Laravel - Visual Studio Code

```
public function update($id,$req $req)
{
    if(Auth::user()->level=="admin"){
        $validator=Validator::make($req->all(),
        [
            'judul'=>'required',
            'penerbit'=>'required',
            'pengarang'=>'required',
            'foto'=>'required'
        ]);
        if($validator->fails()){
            return Response()->json($validator->errors());
        }
        $ubah=Buku_model::where('id',$id)->update([
            'judul'=>$req->judul,
            'penerbit'=>$req->penerbit,
            'pengarang'=>$req->pengarang,
            'foto'=>$req->foto
        ]);
        return Response()->json(['status'=>'berhasil']);
    } else {
        return Response()->json(['status'=>'gagal']);
    }
}

public function destroy($id)
{
    if(Auth::user()->level=="admin"){
        $hapus=Buku_model::where('id',$id)->delete();
        return Response()->json(['status'=>'berhasil']);
    } else {
        return Response()->json(['status'=>'gagal']);
    }
}

public function tampil()
```

```
        'pengarang'=>$req->pengarang,
        'foto'=>$req->foto
    ]);
    return Response()->json(['status'=>'berhasil']);
} else {
    return Response()->json(['status'=>'gagal']);
}

public function destroy($id)
{
    if(Auth::user()->level=="admin"){
        $hapus=Buku_model::where('id',$id)->delete();
        return Response()->json(['status'=>'berhasil']);
    } else {
        return Response()->json(['status'=>'gagal']);
    }
}

public function tampil()
{
    if(Auth::user()->level=="admin"){
        $dt_buku=Buku_model::get();
        return response()->json($dt_buku);
    }else{
        return response()->json(['status'=>'anda bukan admin']);
    }
}
```

## Create Buku

The screenshot shows the Postman application window. The left sidebar displays a list of requests under 'Today'. The main panel shows a POST request to `http://127.0.0.1:8000/api/anggota`. The request body is a JSON object with the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> nama_anggota	Aku aja	
<input checked="" type="checkbox"/> alamat	Malang	
<input checked="" type="checkbox"/> tlp	09876542323	

The response is a JSON object: `{ "status": 1 }`. The status is 200 OK, time is 312ms, and size is 289 B.

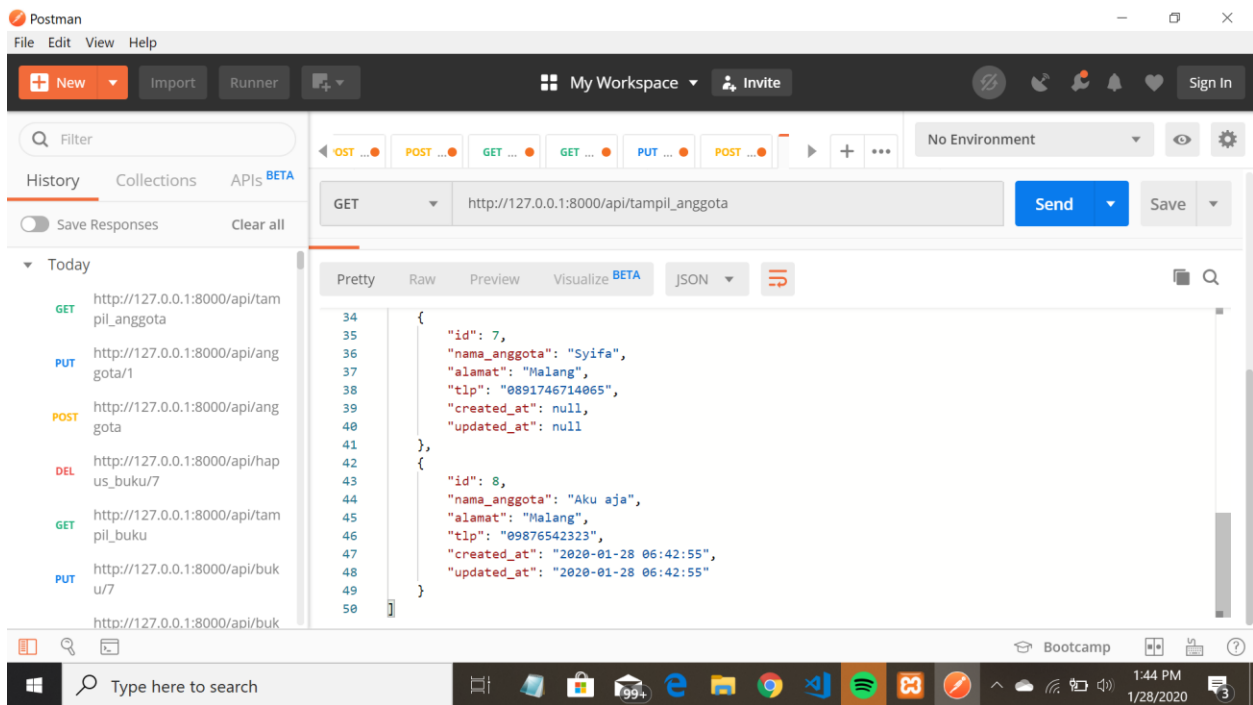
## Update Buku

The screenshot shows the Postman application window. The left sidebar displays a list of requests under 'Today'. The main panel shows a PUT request to `http://127.0.0.1:8000/api/anggota/1`. The request body is a JSON object with the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> nama_anggota	aki aki	
<input checked="" type="checkbox"/> alamat	Malang	
<input checked="" type="checkbox"/> tlp	098765458	

The response is a JSON object: `{ "status": 1 }`. The status is 200 OK, time is 302ms, and size is 289 B.

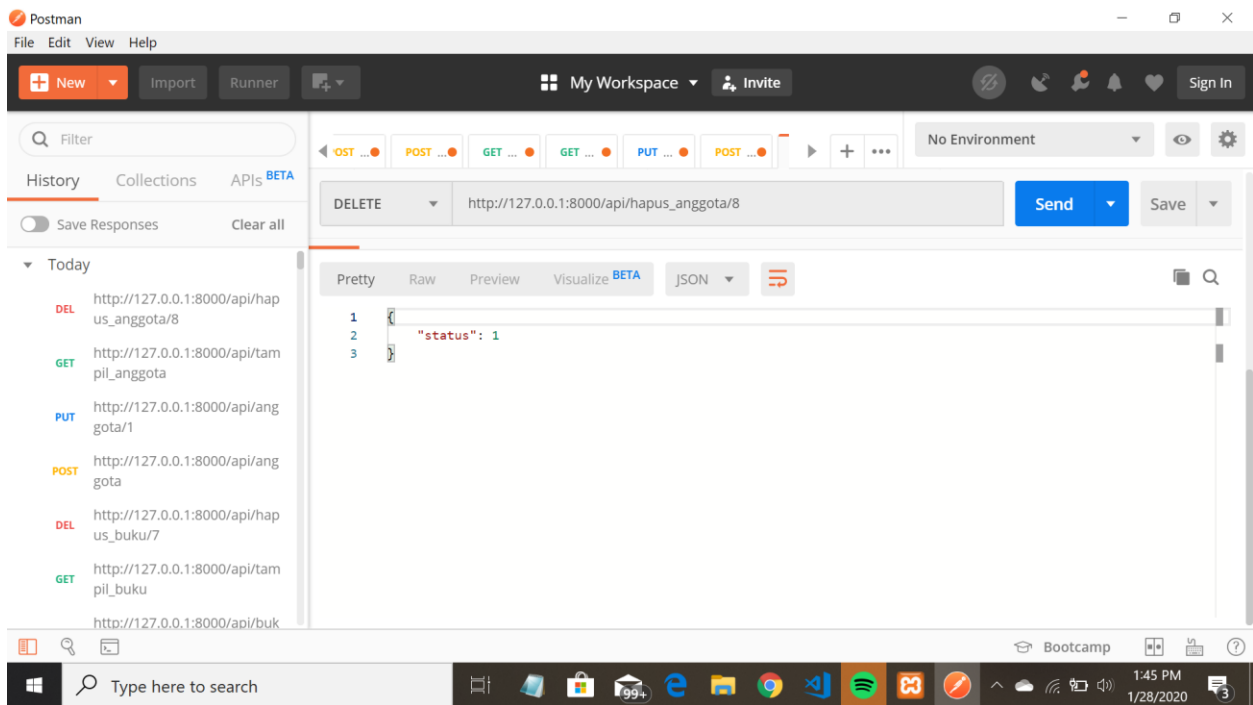
## Tampil buku



Postman interface showing a GET request to `http://127.0.0.1:8000/api/tampil_anggota`. The response is a JSON array of two objects, each representing a member with fields like `id`, `nama_anggota`, `alamat`, `tlp`, `created_at`, and `updated_at`.

```
34 {
35   "id": 7,
36   "nama_anggota": "Syifa",
37   "alamat": "Malang",
38   "tlp": "0891746714065",
39   "created_at": null,
40   "updated_at": null
41 },
42 {
43   "id": 8,
44   "nama_anggota": "Aku aja",
45   "alamat": "Malang",
46   "tlp": "09876542323",
47   "created_at": "2020-01-28 06:42:55",
48   "updated_at": "2020-01-28 06:42:55"
49 }
50 ]
```

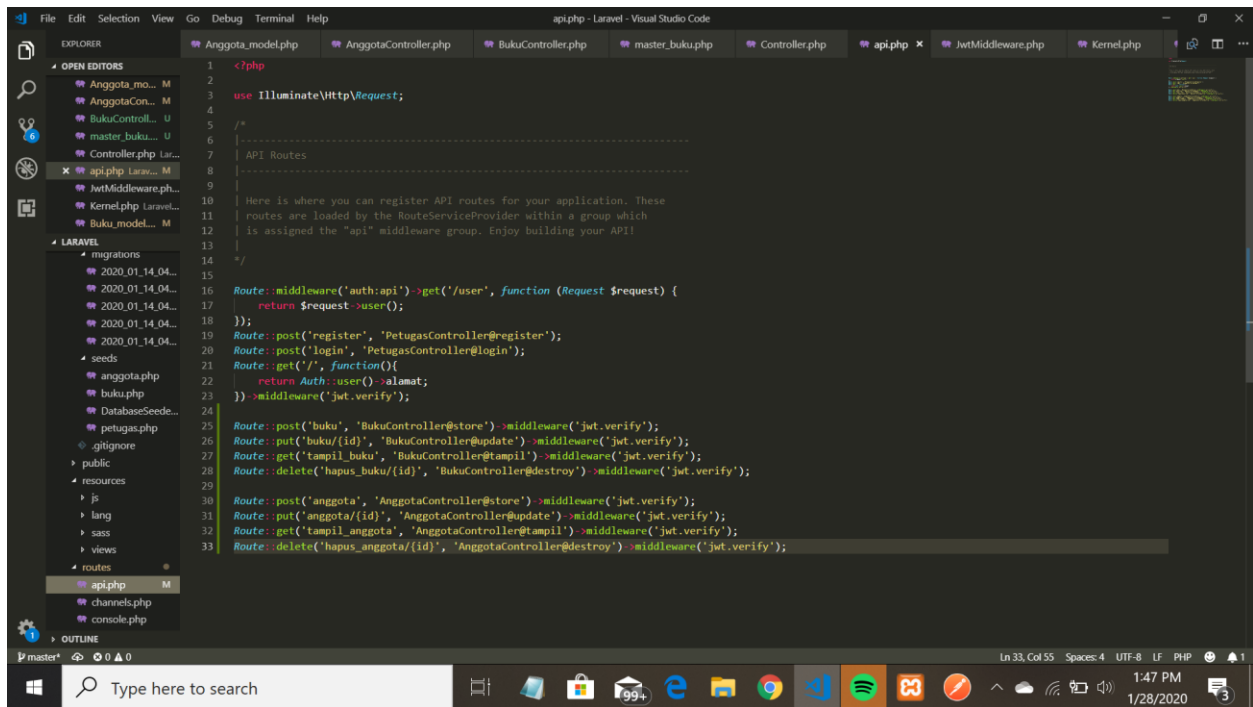
## Hapus Buku



Postman interface showing a DELETE request to `http://127.0.0.1:8000/api/hapus_anggota/8`. The response is a JSON object with the field `status` set to 1.

```
1 {
2   "status": 1
3 }
```

## Api.php



```
1 <?php
2
3 use Illuminate\Http\Request;
4
5 /*
6 |-----
7 | API Routes
8 |-----
9 |
10 | Here is where you can register API routes for your application. These
11 | routes are loaded by the RouteServiceProvider within a group which
12 | is assigned the "api" middleware group. Enjoy building your API!
13 |
14 | */
15
16 Route::middleware('auth:api')->get('/user', function (Request $request) {
17     return $request->user();
18 });
19 Route::post('register', 'PetugasController@register');
20 Route::post('login', 'PetugasController@login');
21 Route::get('/', function() {
22     return Auth::user()->salamat;
23 })->middleware('jwt.verify');
24
25 Route::post('buku', 'BukuController@store')->middleware('jwt.verify');
26 Route::put('buku/{id}', 'BukuController@update')->middleware('jwt.verify');
27 Route::get('tampil_buku', 'BukuController@tampil')->middleware('jwt.verify');
28 Route::delete('hapus_buku/{id}', 'BukuController@destroy')->middleware('jwt.verify');
29
30 Route::post('anggota', 'AnggotaController@store')->middleware('jwt.verify');
31 Route::put('anggota/{id}', 'AnggotaController@update')->middleware('jwt.verify');
32 Route::get('tampil_anggota', 'AnggotaController@tampil')->middleware('jwt.verify');
33 Route::delete('hapus_anggota/{id}', 'AnggotaController@destroy')->middleware('jwt.verify');
```