

LARAVEL 8 REST SERVER API

Pertemuan 1

1. Install Laravel 8
2. Install Jetstream yang livewire , dan sudah termasuk Laravel Sanctum
3. Pergi ke file `app/Http/Kernel.php` tambahkan.

```
'api' => [  
    \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,  
    'throttle:api',  
    \Illuminate\Routing\Middleware\SubstituteBindings::class,  
],
```

4. Pergi ke file `routes/api.php` dan masukan

```
use App\Http\Controllers\API\AuthController;  
  
Route::post('/login',[AuthController::class,'login']);
```

5. Lalu kita buat controllernya

```
php artisan make:controller API/AuthController
```

6. Pergi ke file `App/Http/Controllers/API/AuthController.php` dan masukan codingan.

```

3 namespace App\Http\Controllers\API;
4
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use App\Models\User;
8 use Illuminate\Support\Facades\Hash;
9
10 class AuthController extends Controller
11 {
12     public function login(Request $request)
13     {
14         $user = User::where('email',$request->email)->first();
15         if (!$user || !Hash::check($request->password, $user->password)){
16             return response()->json([
17                 'message' => 'Unauthorized'
18             ],401);
19         }
20         $token = $user->createToken('token-name')->plainTextToken;
21         return response()->json([
22             'message' => 'success',
23             'user' => $user,
24             'token' => $token
25         ],200);
26     }
27 }

```

7. Coba di postman

POST http://127.0.0.1:8000/api/login 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 checked="" type="checkbox"/> | email | rizatsakmir@gmail.com | | | |
| <input checked="" type="checkbox"/> | password | amir00734 | | | |
| | Key | Value | Description | | |

Body Cookies (2) Headers (10) Test Results Status: 200 OK Time: 295 ms Size: 780 B Save Response

Pretty Raw Preview Visualize JSON 🔍

```

3      "user": {
4          "id": 1,
5          "name": "rizat",
6          "email": "rizatsakmir@gmail.com",
7          "email_verified_at": null,
8          "current_team_id": null,
9          "profile_photo_path": "profile-photos/Bqzrcj4hD1X4DnZs1bdDpNvFieGPjHv8AZsueNX.jpg",
10         "created_at": "2021-07-30T12:50:01.000000Z",
11         "updated_at": "2021-07-30T12:51:38.000000Z",
12         "profile_photo_url": "http://localhost:8000/storage/profile-photos/Bqzrcj4hD1X4DnZs1bdDpNvFieGPjHv8AZsueNX.jpg"
13     },
14     "token": "13JhLe4NcMe8rtKjuwKibke6PF8nTxbPzgrIYVtbR1"
15 }

```

Bootcamp Runner Trash

Pertemuan 2

1. Pergi ke file routes/api.php kita tambahkan.

```

use App\Http\Controllers\API\AuthController;
use App\Http\Controllers\API\FormController;

Route::group(['middleware' => 'auth:sanctum'], function() {
    Route::get('/form', [FormController::class, 'index']);
    Route::get('/logout', [AuthController::class, 'logout']);
});

```

2. Kita membuat controller .

```
php artisan make:controller API/FormController
```

3. Pergi ke file App/Http/Controller/API/FormController.php kita tambahkan.
Untuk bisa ke Form Create Data

```
public function index()
{
    return response()->json([
        'message' => 'Ini Form Create Data'
    ],200);
}
```

4. Pergi ke file App/Http/Controller/API/AuthController.php kita tambahkan. Untuk membuat API Logoutnya ,menghapus access tokenya .

```
public function logout(Request $request)
{
    $user = $request->user();
    $user->currentAccessToken()->delete();
    return response()->json([
        'message' => 'Berhasil Logout'
    ],200);
}
```

Nah ini untuk bisa mengakses nya harus menggunakan token yang ada pada saat berhasil login.

GET http://127.0.0.1:8000/api/form Send

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

Type Bearer Token Token 13|JhLe4NcMe8rtKjuwKibke6PF8nTxbPzgr...

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

Body Cookies (2) Headers (10) Test Results Status: 200 OK Time: 233 ms Size: 341 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Ini Form Create Data"
3 }
```

Kalau tidak masukan token tidak bisa mengakses, hanya tampil halaman form login.

GET http://127.0.0.1:8000/api/form Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Type Bearer Token Token

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

Body Cookies (2) Headers (9) Test Results Status: 200 OK Time: 221 ms Size: 4.5 KB Save Response

Pretty Raw Preview Visualize HTML

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <meta name="csrf-token" content="8DM3Z1t8rL4gT90VPhyWiNV5Qf8aPX1sgrMew4YK">
8
9   <title>Laravel</title>
10
11 <!-- Fonts -->
12 <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=swap">
13
```

Pertemuan 3

Create Data

1. Pergi ke halaman routes/api.php dan tambahkan didalam Route Group.

```
Route::post('/create',[FormController::class,'create']);
```

2. Pergi ke file App/Http/Controller/API/FormController.php kita tambahkan.

```
public function create(Request $request)
{
    $request->validate([
        'nama' => 'required|min:5',
        'nim' => 'required|digits:8|numeric',
        'jurusan' => 'required'
    ]);

    $student = new Student;
    $student->nama = $request->nama;
    $student->nim = $request->nim;
    $student->jurusan = $request->jurusan;
    $student->save();

    return response()->json([
        'message' => 'Data Berhasil Disimpan',
        'data' => $student
    ],200);
}
```

3. Membuat model sekaligus migration Student.

```
php artisan make:model Student -m
```

4. Pergi ke file database/migrations/create_students_table.php dan tambahkan .

```
public function up()
{
    Schema::create('students', function (Blueprint $table) {
        $table->id();
        $table->string('nama',40);
        $table->integer('nim'); //minimal 11 character
        $table->string('jurusan',40);
        $table->timestamps();
    });
}
```

5. Pergi ke file app/Models/Student.php dan tambahkan .

```
class Student extends Model
{
    use HasFactory;
    protected $guarded = [];
}
```

6. Coba di postman untuk create data , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .

POST ▼ http://127.0.0.1:8000/api/create 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 checked="" type="checkbox"/> | nama | SanggaYasa | | | |
| <input checked="" type="checkbox"/> | nim | 17190145 | | | |
| <input checked="" type="checkbox"/> | jurusan | Teknologi Informasi | | | |
| | Key | Value | Description | | |

Body Cookies Headers (10) Test Results Status: 200 OK Time: 216 ms Size: 514 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ≡

```

1  {
2    "message": "Data Berhasil Disimpan",
3    "data": {
4      "nama": "SanggaYasa",
5      "nim": "17190145",
6      "jurusan": "Teknologi Informasi",
7      "updated_at": "2021-08-01T14:36:39.000000Z",
8      "created_at": "2021-08-01T14:36:39.000000Z",
9      "id": 5
10   }
11 }

```

Pertemuan 3

Update Data

1. Pergi ke halaman routes/api.php dan tambahkan didalam Route Group.

```
Route::get('/update/{id}',[FormController::class,'getUpdate']);
```

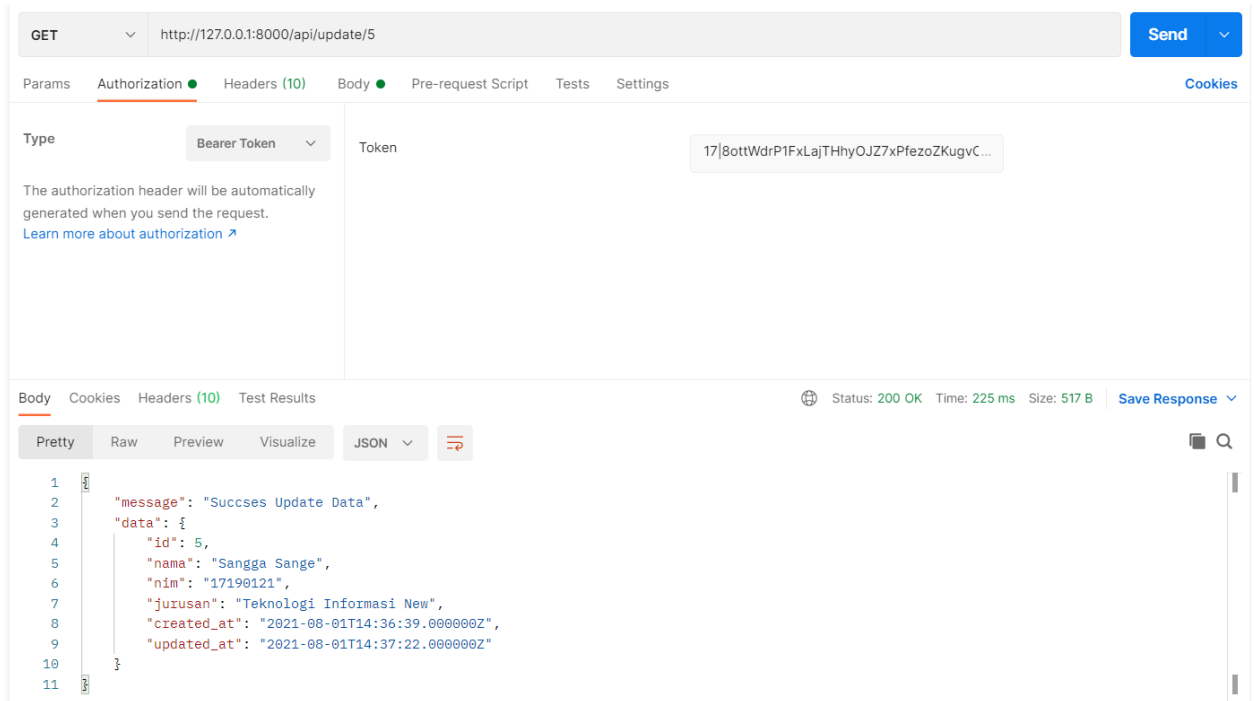
2. Pergi ke file App/Http/Controller/API/FormController.php kita tambahkan.


```

public function getUpdate($id)
{
    $student = Student::find($id);
    if($student != null){
        return response()->json([
            'message' => 'Sukses',
            'data' => $student
        ],200);
    }else{
        return response()->json([
            'message' => 'Data Not Found !',
            'data' => $student
        ],404);
    }
}

```

3. Coba di postman untuk get data update , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .



4. Pergi ke halaman routes/api.php dan tambahkan didalam Route Group.

```
Route::post('/update/{id}',[FormControl::class,'Update']);
```

5. Pergi ke file App/Http/Controller/API/FormController.php kita tambahkan.

```

public function update(Request $request,$id)
{
    $student = Student::find($id);

    $request->validate([
        'nama' => 'required|min:5',
        'nim' => 'required|digits:8|numeric',
        'jurusan' => 'required'
    ]);

    $student->update([
        'nama' => $request->nama,
        'nim' => $request->nim,
        'jurusan' => $request->jurusan
    ]);

    return response()->json([
        'message' => 'Succses Update Data',
        'data' => $student
    ],200);
}

```

6. Coba di postman untuk update data, masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .

POST http://127.0.0.1:8000/api/update/5 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 checked="" type="checkbox"/> | nama | Sangga Sange | | | |
| <input checked="" type="checkbox"/> | nim | 17190121 | | | |
| <input checked="" type="checkbox"/> | jurusan | Teknologi Informasi New | | | |
| | Key | Value | Description | | |

Body Cookies Headers (10) Test Results Status: 200 OK Time: 225 ms Size: 517 B Save Response

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "Sukses Update Data",
3    "data": {
4      "id": 5,
5      "nama": "Sangga Sange",
6      "nim": "17190121",
7      "jurusan": "Teknologi Informasi New",
8      "created_at": "2021-08-01T14:36:39.000000Z",
9      "updated_at": "2021-08-01T14:37:22.000000Z"
10   }
11 }

```

Pertemuan 4

Delete Data

1. Pergi ke halaman routes/api.php dan tambahkan didalam Route Group.

```
Route::get('/delete/{id}', [FormController::class, 'delete']);
```

2. Pergi ke file App/Http/Controller/API/FormController.php kita tambahkan.

```

public function delete($id)
{
    $student = Student::find($id);
    if($student != null){
        Student::find($id)->delete();
        return response()->json([
            'data' => $student,
            'message' => 'Data Berhasil Di Hapus'
        ],200);
    }else{
        return response()->json([
            'message' => 'Data Not Found'
        ],404);
    }
}
}

```

3. Coba di postman untuk delete data , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .

http://127.0.0.1:8000/api/delete/5

GET http://127.0.0.1:8000/api/delete/5 Send

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 |
|-----|-------|-------------|-----|-----------|
| Key | Value | Description | | |

Body Cookies Headers (10) Test Results Status: 200 OK Time: 213 ms Size: 518 B Save Response

Pretty Raw Preview Visualize JSON

```

1  {
2    "data": {
3      "id": 5,
4      "nama": "Sangga Sange",
5      "nim": 17190121,
6      "jurusan": "Teknologi Informasi New",
7      "created_at": "2021-08-01T14:36:39.000000Z",
8      "updated_at": "2021-08-01T14:37:22.000000Z"
9    },
10   "message": "Data Berhasil Di Hapus"
11 }

```

Pertemuan 5

Create Data Multiple

1. Kita membuat model dan migration dengan nama score .

```
php artisan make:model score -m
```

2. Pergi ke file database/migrations/create_students_table.php dan tambahkan .

```

public function up()
{
    Schema::create('scores', function (Blueprint $table) {
        $table->id();
        $table->integer('student_id');
        $table->string('mata_pelajaran',50);
        $table->integer('nilai');
        $table->timestamps();
    });
}

```

3. Pergi ke file app/Models/Student.php dan tambahkan .

```

class Score extends Model
{
    use HasFactory;

    protected $guarded = [];
}

```

4. Kita migrate , “php artisan migrate”.

5. Membuat ScoreController .

```

php artisan make:controller API/ScoreController

```

6. Pergi ke file routes/api.php dan tambahkan ini.

```

use App\Http\Controllers\API\ScoreController;

```

```
// CRUD MultiTable  
Route::post('/create_score_data',[ScoreController::class,'create']);
```

7. Pergi ke file `app/Http/Controllers/API/ScoreController.php` dan tambahkan.


```

public function create(Request $request)
{
    $request->validate([
        'nama' => 'required|min:5',
        'nim' => 'required|digits:8|numeric',
        'jurusan' => 'required'
    ]);

    $student = new Student;

    $student->nama = $request->nama;

    $student->nim = $request->nim;

    $student->jurusan = $request->jurusan;

    $student->save();

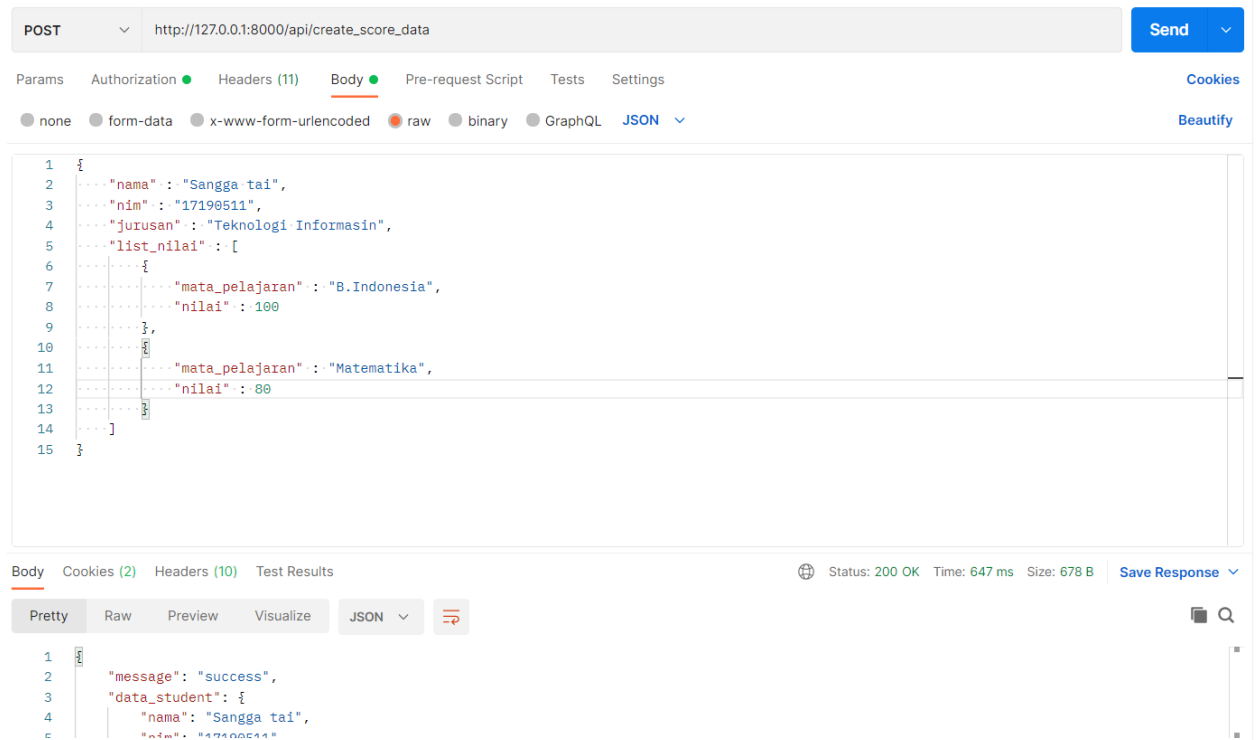
    foreach($request->list_nilai as $nilai => $value){
        $score = array(
            'student_id' => $student->id,
            'mata_pelajaran' => $value['mata_pelajaran'],
            'nilai' => $value['nilai']
        );

        $score = Score::create($score);
    }

    return response()->json([
        'message' => 'success',
        'data_student' => $student,
        'nilai_student'=> $score
    ],200);
}

```

4. Coba di postman untuk create data multiple , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .



PERTEMUAN 6

UPDATE DATA RELASIONSHIP

1. Pergi ke file routes/api.php tambahkan ini .

```
Route::get('/update_score_dataStudent/{id}',[ScoreController::class,'getUpdate']);
```

2. Pergi ke app/Http/Models/Student.php tambahkan ini . untuk relasi join , hasMany = OneToMany.

```

class Student extends Model
{
    use HasFactory;
    protected $guarded = [];

    public function score()
    {
        return $this->hasMany('App\Models\Score', 'student_id');
    }
}

```

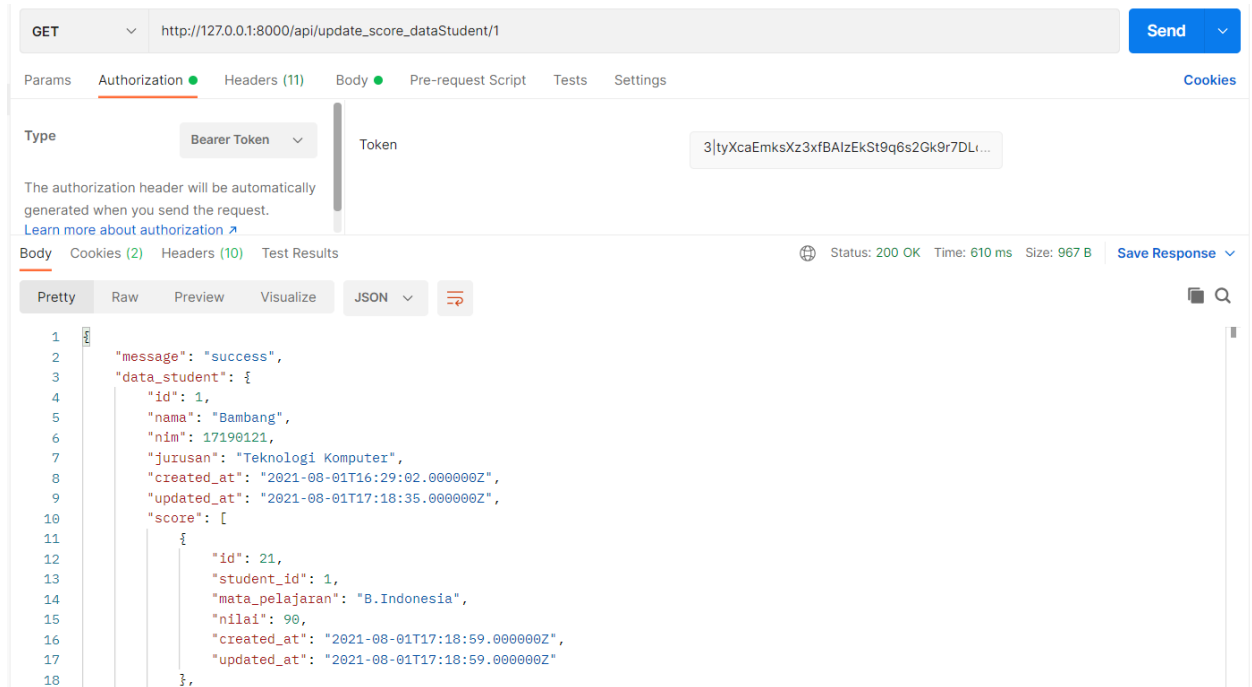
3. Pergi ke file `app/Http/Controllers/API/ScoreController.php` dan tambahkan ini .

```

public function getUpdate($id)
{
    $student = Student::with('score')->where('id',$id)->first();
    return response()->json([
        'message' => 'success',
        'data_student' => $student
    ],200);
}

```

4. Coba di postman untuk get data update multiple , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Application/json' .



5. Pergi ke file routes/api.php tambahkan ini .

```
Route::post('/update_score_dataStudent/{id}',[ScoreController::class,'update']);
```

6. Pergi ke file app/Http/Controllers/API/ScoreController.php dan tambahkan ini .

```
public function update(Request $request, $id)
{
    $student = Student::find($id);
    $student->update([
        'nama' => $request->nama,
        'nim' => $request->nim,
        'jurusan' => $request->jurusan
    ]);

    Score::where('student_id',$id)->delete();

    foreach($request->list_nilai as $nilai => $value){
        $score = array(
            'student_id' => $id,
            'mata_pelajaran' => $value['mata_pelajaran'],
            'nilai' => $value['nilai']
        );
        $score = Score::create($score);
    }

    return response()->json([
        'message' => 'success',
        'data_student' => $student,
        'nilai_student'=> $score
    ],200);
}
```

7. Coba di postman untuk update data multiple , masukan tokenya di Authorization dengan type Bearer Token , dan masukan di Headers yaitu 'Accept : Aplication/json' .

The screenshot displays the Postman interface for a REST client. The top section shows a POST request to the endpoint `http://127.0.0.1:8000/api/update_score_dataStudent/1`. The 'Body' tab is selected, showing a JSON payload for updating a student's scores. The payload includes the student's name, NIM, faculty, and a list of subjects with their respective scores. The bottom section shows the response, which is a JSON object indicating a successful update with a message and the updated student data.

Request:

```
1 {
2   "nama": "Bambang",
3   "nim": "17190121",
4   "jurusan": "Teknologi Komputer",
5   "list_nilai": [
6     {
7       "mata_pelajaran": "B.Indonesia",
8       "nilai": 90
9     },
10    {
11      "mata_pelajaran": "Matematika",
12      "nilai": 75
13    },
14    {
15      "mata_pelajaran": "B.Ingggris",
16      "nilai": 55
17    }
18  ]
19 }
```

Response:

```
1 {
2   "message": "success",
3   "data_student": {
4     "id": 1,
```

Status: 200 OK | **Time:** 610 ms | **Size:** 967 B | [Save Response](#)

SELESAI