REST API CRUD Using Model in LARAVEL

MIGRATION Page –

Command :

php artisan make:migration create_students_table

php artisan migrate

php artisan migrate:refresh

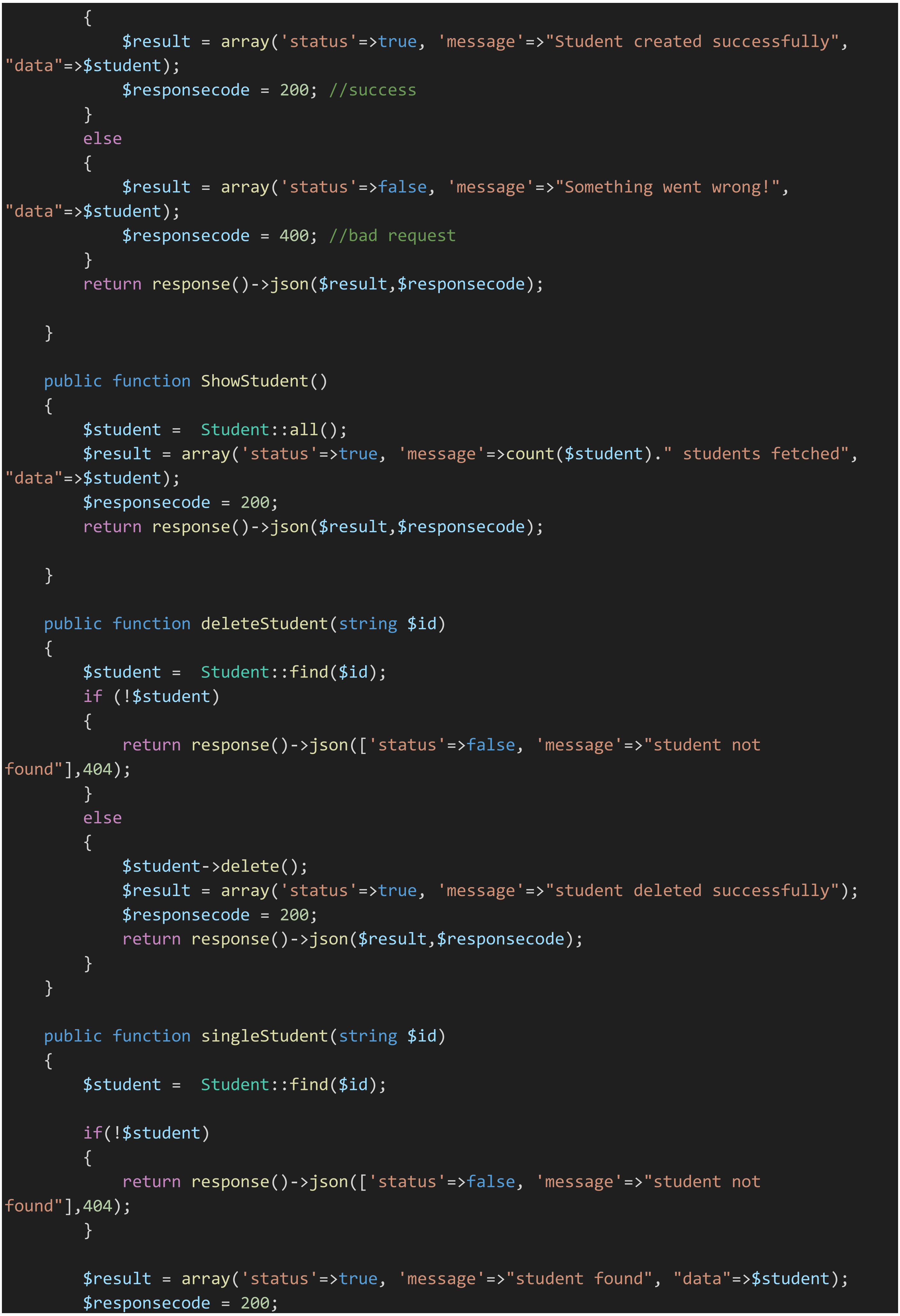
```php
<?php

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('students', function (Blueprint $table) {
            $table->integer('id');
            $table->string('fname',20);
            $table->string('lname',20);
            $table->string('email',50);
            $table->string('password',30);
            $table->string('confpassword',30);
            $table->string('class',10);
            $table->string('gender',10);
            $table->text('address');
            $table->string('phone',15);
            $table->timestamps();
        });
    }

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

Controller Page –

Command:

php artisan make:controller StudentController

```
<?php

namespace App\Http\Controllers\API;

use App\Http\Controllers\Controller;
use App\Models\Student;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;

class StudentController extends Controller
{
    public function addStudent(Request $req)
    {
        $validator = Validator::make($req->all(),[
            'fname'=>'required',
            'lname'=>'required',
            'email'=>'required|email',
            'password'=>'required|min:8',
            'confpassword'=>'required|same:password|min:8',
            'gender'=>'required',
            'address'=>'required',
            'class'=>'required',
            'phone'=>'required|numeric|digits:10',

        ]);
        if($validator->fails())
        {
            $result = array('status' => true, 'message' =>'Validation error 
occurred','error_message' => $validator->errors());
            return response()->json($result,400);
        }
        $student = Student::create(
                    [

                        'id'=>rand(111,999999999),
                        'fname'=>$req->fname,
                        'lname'=>$req->lname,
                        'email'=>$req->email,
                        'password'=>$req->password,
                        'confpassword'=>$req->confpassword,
                        'gender'=>$req->gender,
                        'address'=>$req->address,
                        'class'=>$req->class,
                        'phone'=>$req->phone,

                    ]
                    );

        if ($student)
```



```
        {
            $result = array('status'=>true, 'message'=>"Student created successfully",
"data"=>$student);
            $responsecode = 200; //success
        }
        else
        {
            $result = array('status'=>false, 'message'=>"Something went wrong!",
"data"=>$student);
            $responsecode = 400; //bad request
        }
        return response()->json($result,$responsecode);

    }

    public function ShowStudent()
    {
        $student =  Student::all();
        $result = array('status'=>true, 'message'=>count($student)." students fetched",
"data"=>$student);
        $responsecode = 200;
        return response()->json($result,$responsecode);

    }

    public function deleteStudent(string $id)
    {
        $student =  Student::find($id);
        if (!$student)
        {
            return response()->json(['status'=>false, 'message'=>"student not
found"],404);
        }
        else
        {
            $student->delete();
            $result = array('status'=>true, 'message'=>"student deleted successfully");
            $responsecode = 200;
            return response()->json($result,$responsecode);
        }
    }

    public function singleStudent(string $id)
    {
        $student =  Student::find($id);

        if(!$student)
        {
            return response()->json(['status'=>false, 'message'=>"student not
found"],404);
        }

        $result = array('status'=>true, 'message'=>"student found", "data"=>$student);
        $responsecode = 200;
        return response()->json($result,$responsecode);


    }

    public function updateStudent(Request $req , $id)
    {
        $student =  Student::find($id);
        if(!$student)
        {
            return response()->json(['status'=>false, 'message'=>"student not
found"],404);
        }

        $validator = validator::make($req->all(),
        [
            'fname'=>'required',
            'lname'=>'required',
            'email'=>'required|email',
            'password'=>'required|min:8',
            'confpassword'=>'required|same:password|min:8',
            'gender'=>'required',
            'address'=>'required',
            'class'=>'required',
            'phone'=>'required|numeric|digits:10',

        ]);
        if($validator->fails())
        {
            $result = array('status' => true, 'message' =>'Validation error
occurred','error_message' => $validator->errors());
            return response()->json($result,400);
        }
        $student->update(
            [

                        'id'=>rand(111,999999999),
                        'fname'=>$req->fname,
                        'lname'=>$req->lname,
                        'email'=>$req->email,
                        'password'=>$req->password,
                        'confpassword'=>$req->confpassword,
                        'gender'=>$req->gender,
                        'address'=>$req->address,
                        'class'=>$req->class,
                        'phone'=>$req->phone,

            ]);

        if ($student)
        {
            $result = array('status'=>true, 'message'=>"Student updated successfully",
"data"=>$student);
            $responsecode = 200; //success
        }
        else
        {
            $result = array('status'=>false, 'message'=>"Something went wrong!",
"data"=>$student);
            $responsecode = 400; //bad request
        }
        return response()->json($result,$responsecode);

    }
}
```

MODEL Page –

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Student extends Model
{
    use HasFactory;

    protected $table = 'Students';

    protected $fillable = [
        'id',
        'fname',
        'lname',
        'email',
        'password',
        'confpassword',
        'class',
        'gender',
        'address',
        'phone',

    ];

}
```

ROUTE Page –

```
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\http\Controllers\API\StudentController;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "api" middleware group. Make something great!
|
*/

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});

route::post('/add-student',[StudentController::class,'addStudent']);

route::get('/single-student/{id}',[StudentController::class,'singleStudent']);

route::get('/show-student',[StudentController::class,'ShowStudent']);

route::delete('/delete-student/{id}',[StudentController::class,'deleteStudent']);

route::post('/update-student/{id}',[StudentController::class,'updateStudent']);
```

Call api in postman urls

Add url (post)

http://127.0.0.1:8000/api/add-student

show url (get)

http://127.0.0.1:8000/api/show-student

show-single-data url (get)

http://127.0.0.1:8000/api/single-student/758247697

Delete url (delete)

http://127.0.0.1:8000/api/delete-student/341597153

Update url (post/put)

http://127.0.0.1:8000/api/update-student/16263201