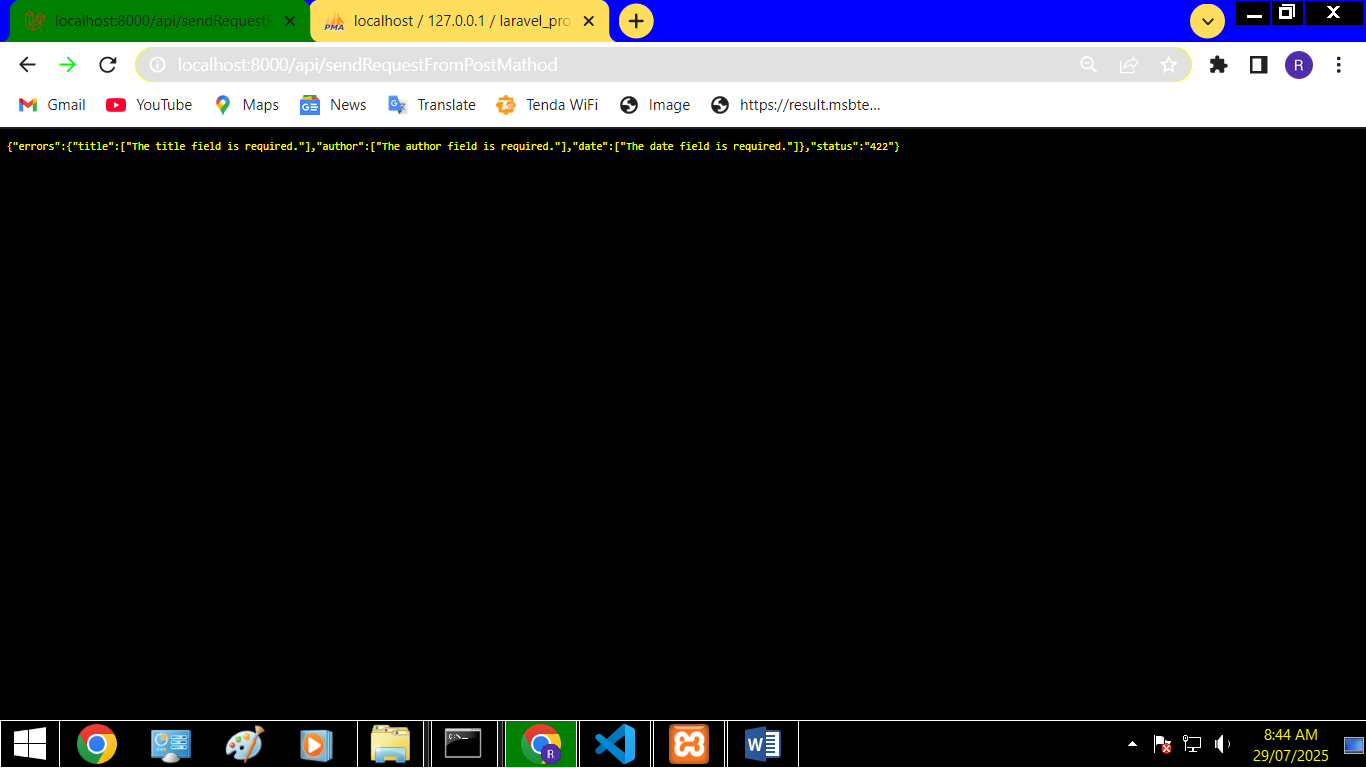
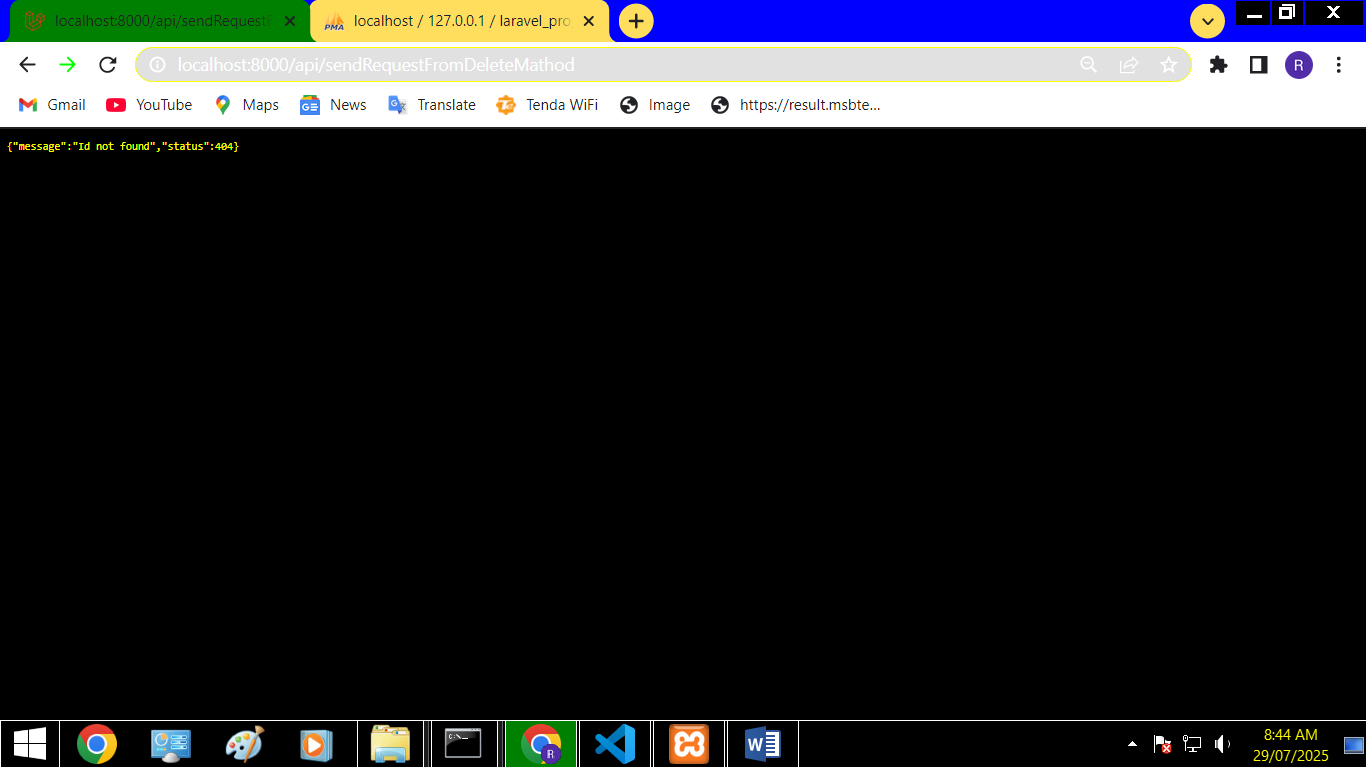


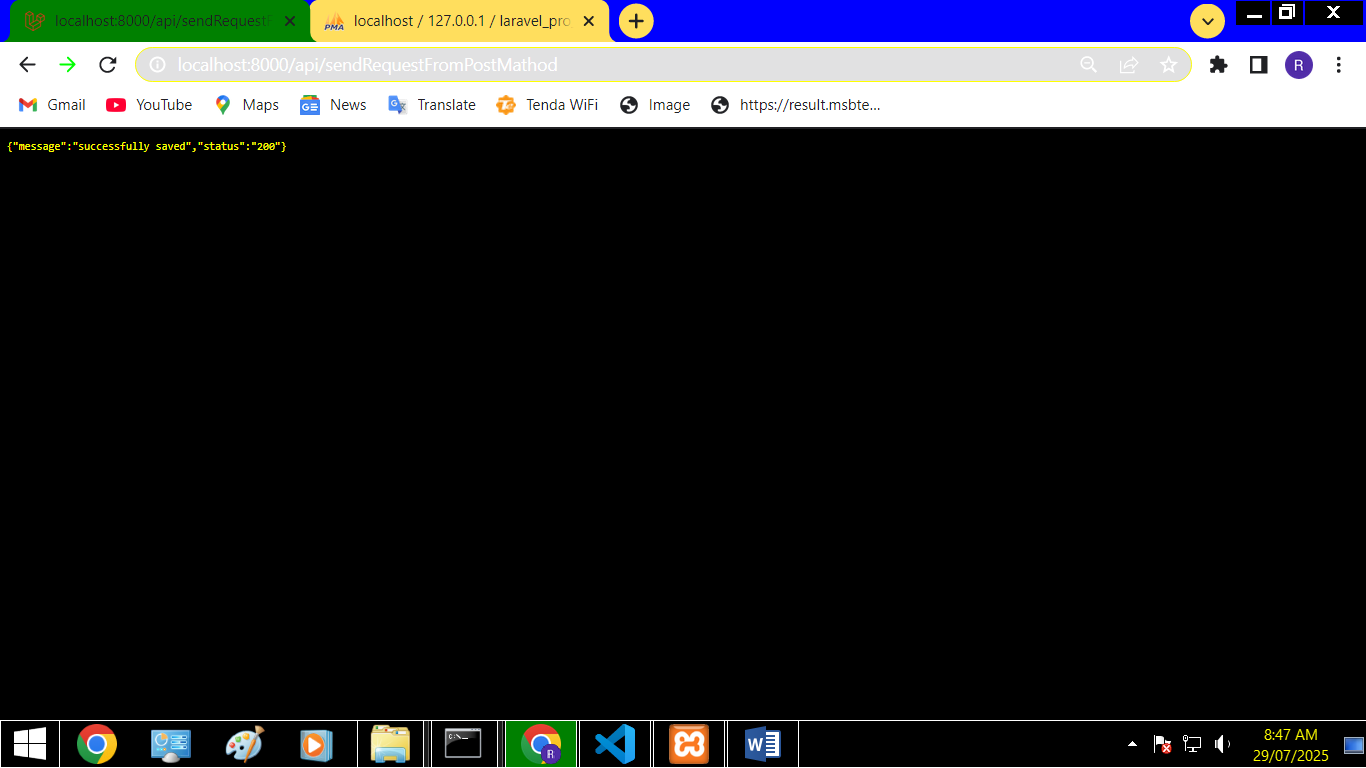
Errors:



Both for Update and Delete:



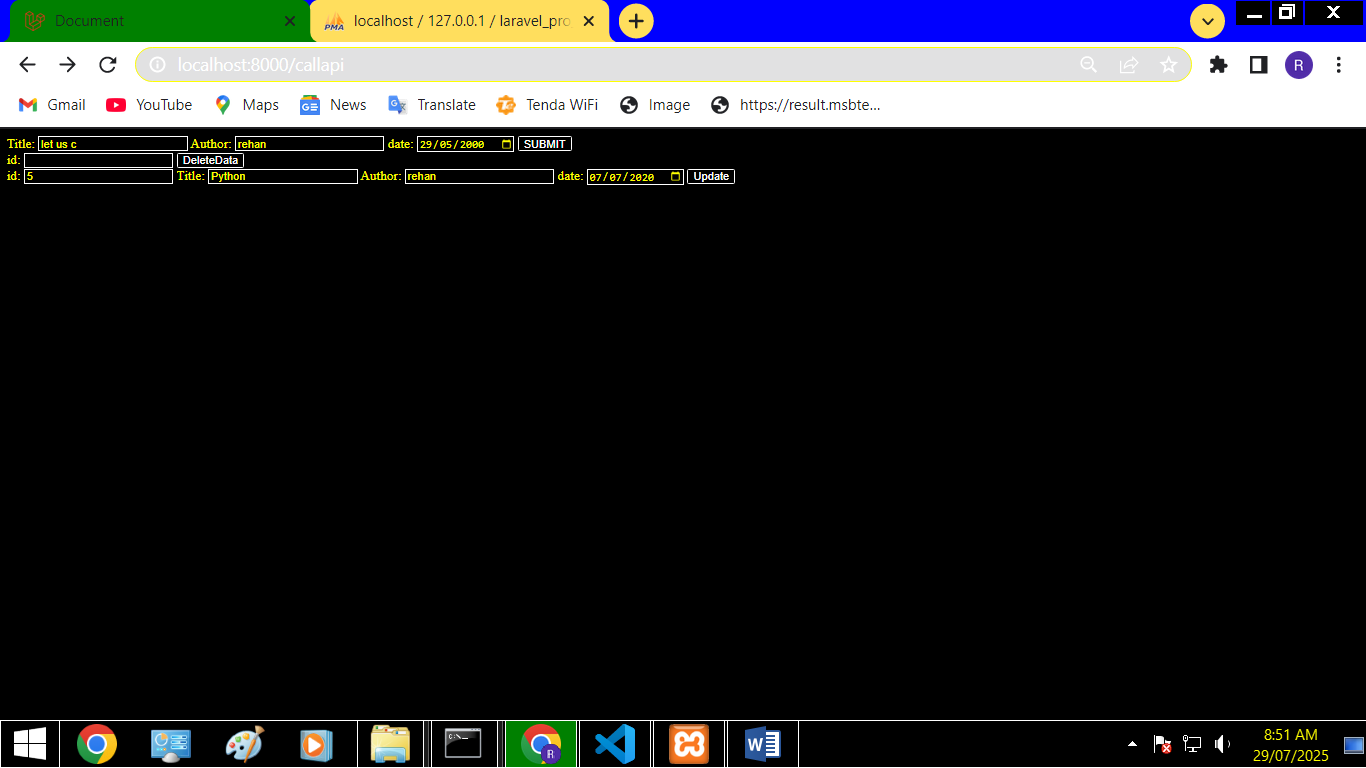
Insert , Update ,Delete:

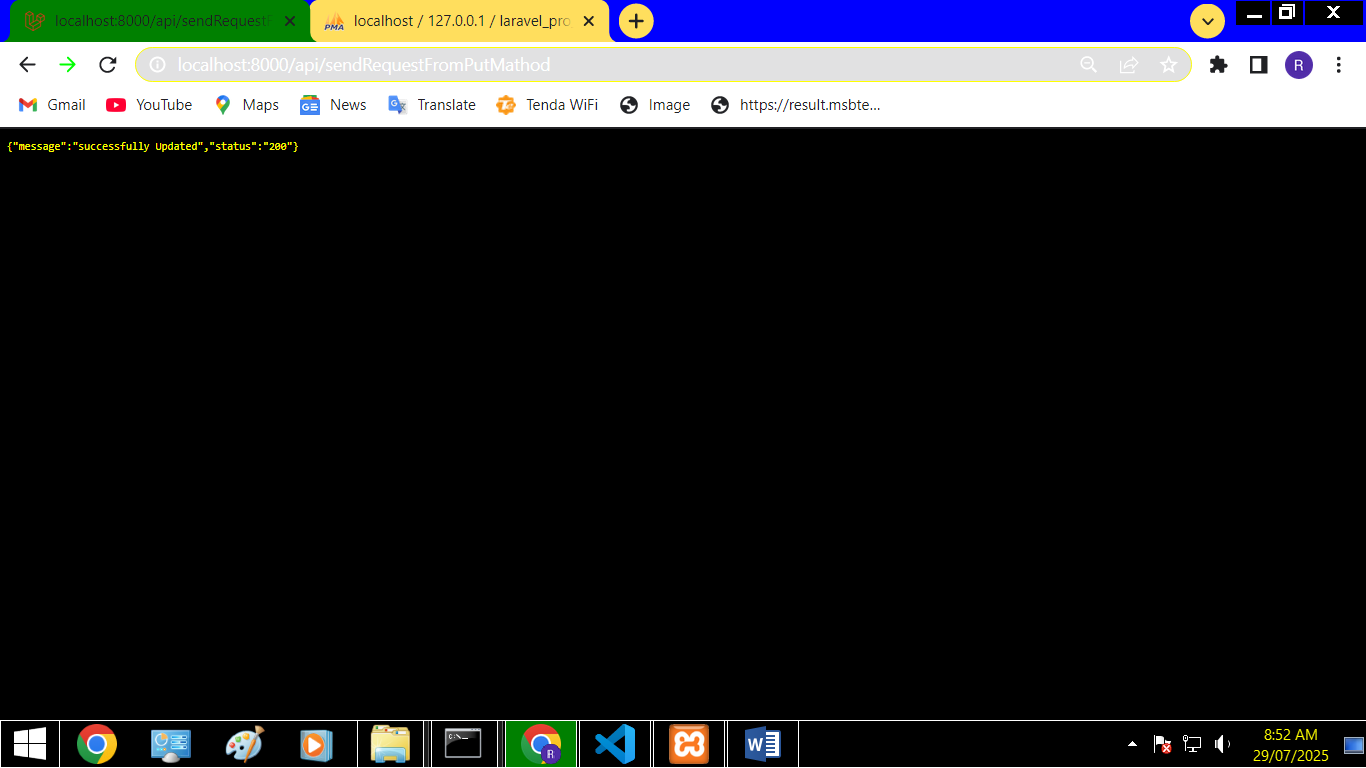


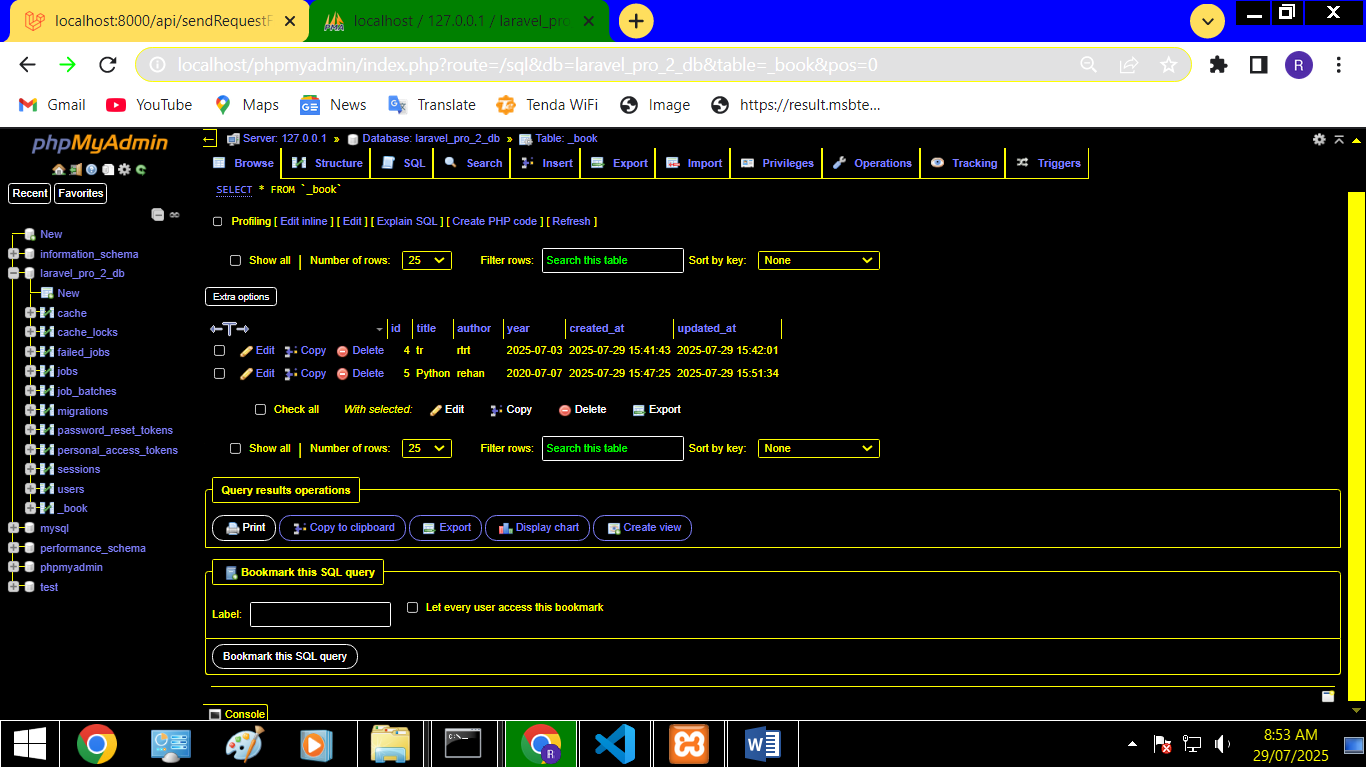


inserted

Update:

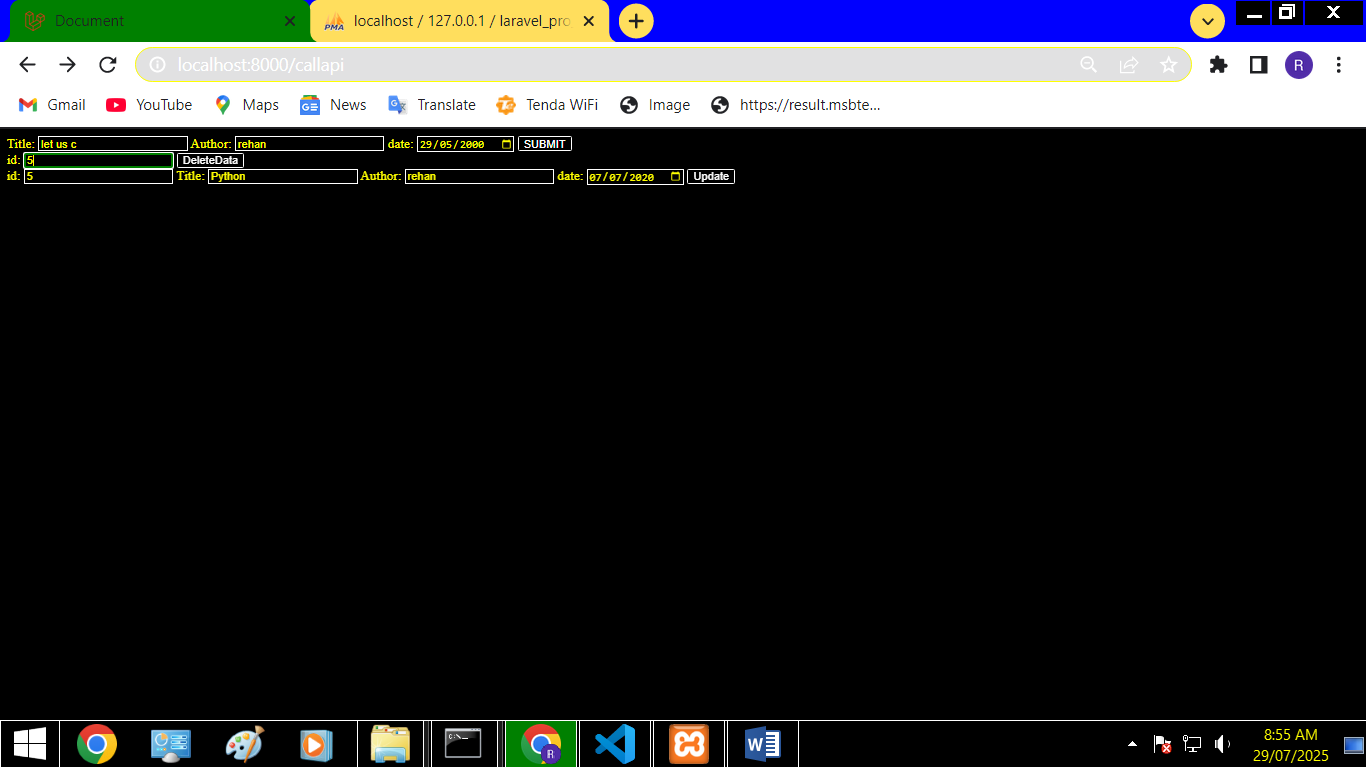


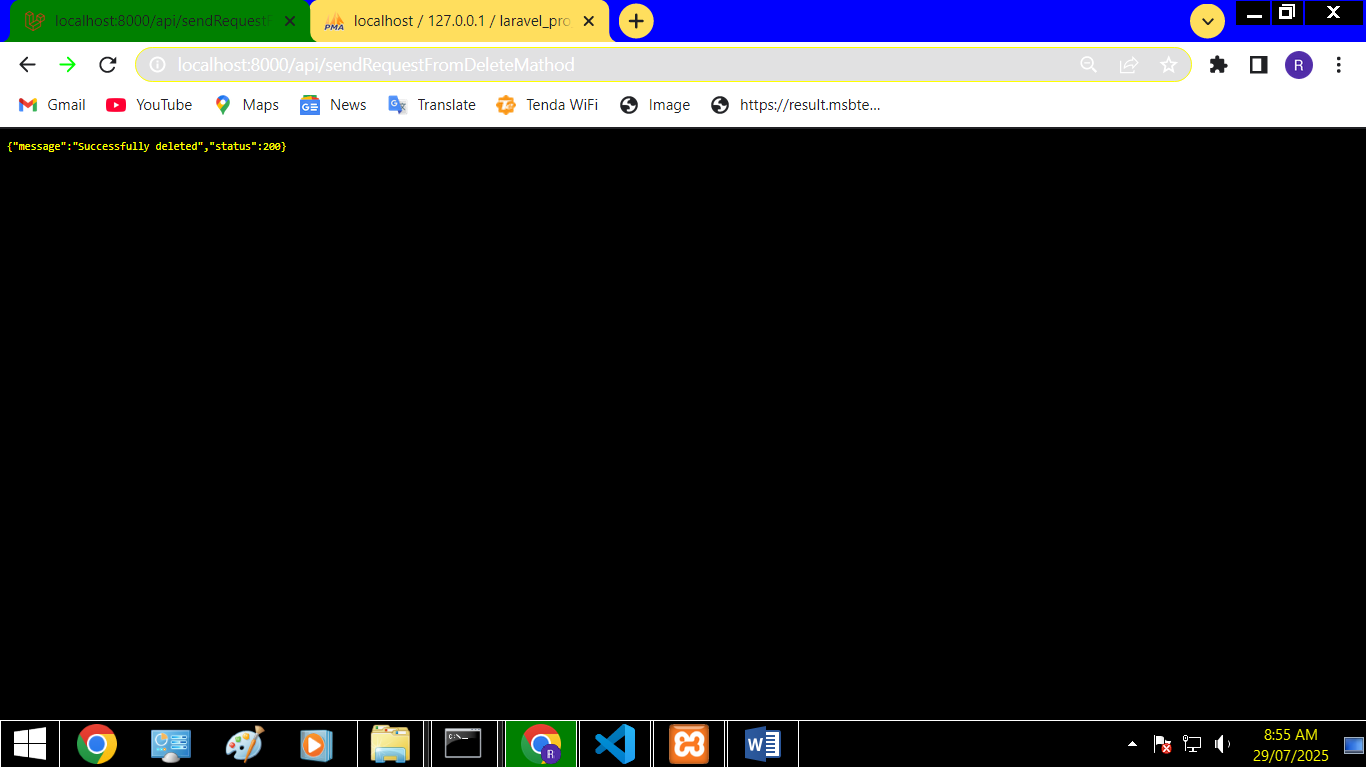




updated

Delete:





Balde file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <form action="api\sendRequestFromPostMathod" method="post">

        @csrf

   <label>

    Title:

    <input type="text" name="title">

</label>

<label>

    Author:

    <input type="text" name=" author">

</label>

<label>

    date:

    <input type="date" name="date">

</label>

   <button type="submit">SUBMIT</button>

    </form>

    <form action="api\sendRequestFromDeleteMathod" method="post">

    @csrf

    @method('DELETE')

<label>

    id:

    <input type="text" name="id">

</label>

     <button type="submit">DeleteData</button>

</form>

<form action="api\sendRequestFromPutMathod" method="post">

        @csrf

        @method('PUT')

<label>

    id:

    <input type="text" name="id">

</label>

   <label>

    Title:

    <input type="text" name="title">

</label>

<label>

    Author:

    <input type="text" name=" author">

</label>

<label>

    date:

    <input type="date" name="date">

</label>

   <button type="submit">Update</button>

    </form>

   <p>@error('title') {{ $message }} @enderror</p>

</body>

</html>

Web file:

Route::view('callapi','testApiFromPostMathod');

Api.file:

Route::post('sendRequestFromPostMathod', [con1::class, 'savedata']);

Route::delete('sendRequestFromDeleteMathod' , [con1::class, 'delete']);

Route::put('sendRequestFromPutMathod' , [con1::class, 'put']);

Controller file:

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

use App\Models\Book;

class con1 extends Controller

{

    function savedata(Request $req) {

        $validated = Validator::make($req->all(), [

            'title' => 'required|string',

            'author' => 'required|string',

            'date' => 'required|date'

        ]);

        if ($validated->fails()) {

            return response()->json([

                'errors' => $validated->errors(),

                'status' => '422'

            ], 422);

        } else {

             $obj = new Book();

             $obj->title = $req->title;

             $obj->author = $req->author;

             $obj->year = $req->date;

             $obj->save();

            return response()->json([

                'message' => 'successfully saved',

                'status' => '200'

            ], 200);

        }

    }

    function delete(Request $req) {

        $id = $req->id;

        $data = new Book;

        $finaldata = $data->find($id);

        if($finaldata != null) {

            $finaldata->delete();

            return response()->json([

                'message' => 'Successfully deleted',

                'status' => 200

            ], 200);

        } else {

            return response()->json([

                'message' => 'Id not found',

                'status' => 404

            ], 404);

        }

    }

    function put(Request $req) {

        $validated = Validator::make($req->all(), [

            'id'=> 'required',

            'title' => 'required|string',

            'author' => 'required|string',

            'date' => 'required|date'

        ]);

        if ($validated->fails()) {

            return response()->json([

                'errors' => $validated->errors(),

                'status' => '422'

            ], 422);

        } else {

               $id = $req->id;

             $obj =  Book::find($id);

            if($obj != null){

             $obj->title = $req->title;

             $obj->author = $req->author;

             $obj->year = $req->date;

             $obj->save();

            return response()->json([

                'message' => 'successfully Updated',

                'status' => '200'

            ], 200);

        }  else {

            return response()->json([

                'message' => 'Id not found',

                'status' => 404

            ], 404);

        }

        }

    }

}

/\*if($datafi == null) {

    $datafi->delete();

     return response()->json([

    'message' => 'successfully deleted',

    'status' => '200'], 200);

     } else {

        return response()->json([

            'message' => 'Id not found',

            'status' => '400'], 400);

     }\*/