<h2>ভূমিকার আগে</h2>

<p>আমি এই লিখায় লারাভেলের Facade কি এবং কিভাবে তার উত্তর দিতে চেষ্টা করব। তবে কেউ যদি জানতে চান Facade কেন তাহলে একটু কষ্ট করে <a href="<https://laravel.com/docs/5.5/facades>" target="\_blank">লারাভেলের ডকুমেন্টেশনের Facade-সেকশনটা</a> দেখে নিতে পারেন। যদি ডকুমেন্টেশনের কোন অংশ বুঝতে অসুবিধা হয় তাহলে নিঃসংকোচে আমাকে প্রশ্ন করতে পারেন। আমি আজকের লিখায় লারাভেল Facade কিভাবে কাজ করে? কিভাবে নিজের মত করে নিজের একটি Facade বানিয়ে ফেলা যায়? এই সব প্রশ্নের উত্তর দেওয়ার চেষ্টা করব এই লিখায়। </p>

<h2>ভূমিকা</h2>

<p>লারাভেলে কাজ করার শুরুর দিকে আমি Facade-কে ভাবতাম এমন একটা ক্লাস যার মধ্যে কতগুলো স্টেটিক ম্যাথড আছে। এর পিছনে কারণ হচ্ছে লারাভেল যারা শুরু করে তাদের কাছে লারাভেলের এপ্লিকেশনে ঢোকার রাস্তা(Entry point) হলো রুট(route.php বর্তমানে web.php, api.php ইত্যাদি) ফাইল। আর সেই রুট ফাইলে আমরা কোন রুট লিখি এভাবে :</p>

<pre><code>Route::get('home', 'HomeController@show');</code></pre>

<p>কেবল মাত্র প্রাথমিক অবজেক্ট অরিয়েন্টেড জ্ঞান সম্পন্ন যে কোন বিশ্বাবিদ্যালয় পড়ুয়া প্রোগ্রামার উপরের এই লাইনটা দেখে এটা ভাবাই স্বাভাবিক যে, নিশ্চই Route নামে একটা ক্লাস আছে যার মধ্যে get নামে একটা স্টেটিক মেথড আছে। আর এখানে সেই মেথডটিতে দুইটা আরগুমেন্ট পাঠানো হচ্ছে। অন্য কেউ শুরুর দিকে ভাবুব-না ভাবুক আমি অন্তত্য তাই ভাবছিলাম। তো কিছুদিন পর আমি কৌতুহলবসত দেখতে চাইলাম যে এই get মেথডটিতে কি লিখা আছে। কিন্তু আমি আমার কোড এডিটর থেকে যখন ঐ Route ক্লাসটিতে গেলাম তখন বেশ বড় ধরনের একটা ধাক্কা খেলাম কারণ ঐ ক্লাসে get নামে কোন মেথড নেই! তাহলে কি আমি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর এমন সাধারণ বিষয়ও ঠিকমত জানি না? পরে জানতে পারলাম এই Route ক্লাসটি লারাভেলের এক ধরনের বিশেষ ক্লাস যাদেরকে আসলে Facade বলা হয়। এই Facade নিয়েই আজকের লিখা।</p>

<p>পরবর্তিতে, আমি যখন আমাদের কোম্পানিতে নিয়োগের জন্য লারাভেলে অভিজ্ঞতাসম্পন্ন বিভিন্ন প্রোগ্রামাদের সাক্ষাৎকার নিতাম তখন এই প্রশ্নটা প্রায়ই করতাম। কিন্তু দুঃখের বিষয় হল, অনেক প্রোগ্রামারকে আমি স্টেটিক মেথড কি এটা নিয়েই দ্বিধায় ভুগতে দেখেছি। আর যাদের স্টেটিক মেথড সম্পর্কে ধারণ আছে তারা এই ব্যাপারে আমার সাথে এক মত হতেন যে, Route ক্লাসের মধ্যে get নামে একটি স্টেটিক মেথড আছে, যা আসলে ভূল। এই ভূল উত্তরটি আমি আনাড়ী লারাভেল প্রোগ্রামার থেকে শুরু করে দুই তিন বছরের অভিজ্ঞতা সম্পন্ন প্রগ্রামারদের কাছেও পেয়েছি, যা আসলে একেবারেই অপ্রত্যাশিত ছিল। </p><br>

<h2>Facade কি? </h2>

<p>লারাভেলের Facade হল এমন একটি ক্লাস যে অন্য একটি ক্লাসের স্টেটিক ইন্টারফেইস হিসাবে কাজ করে। আরেকটু সহজভাবে বললে, Facade ক্লাস অন্য একটি ক্লাসের মেথড কে কল করার মাধ্যম হিসাবে কাজ করে। তবে ঐ ক্লাসটির কোন মেথড স্টেটিক মেথড না হওয়া সত্যেও Facade ক্লাস ব্যাবহার করে যখন কল করা হয় স্টেটিক মেথডের মত কল করা যায়। তার মানে, এখন আমরা বলতে পারি, লারাভেলের Facade আসলে অন্য একটি ক্লাসের নন-স্টেটিক মেথডকে স্টেটিক মেথডের মত কল করার মাধ্যম হিসাবে কাজ করে। Route এর মত লারাভেলে আরও অনেক জনপ্রিয় Facade আছে যেমন, View, Session, Cache, Request ইত্যাদি</p><br>

<h2>কিভাবে নিজস্ব Facade বানাতে হয়? </h2>

<p>উদাহরণের স্বার্থে মনে করি, আমরা আমাদের লারাভেল এপ্লিকেশনে গানিতিক হিসাব-নিকাশ করার জন্য Math একটি নামে Facade লিখতে চাই। যেই Facade ব্যাবহার করে, Math::add($number1, $number2) ও Math::sub($number1, $number2) লিখলে যথাক্রমে দুইটি নাম্বারের যোগ ও বিয়োগ করা যাবে। এর জন্য নিচের ৩টি ধাপ অনুসরণ করতে হবেঃ </p>

<p><strong>১।</strong> একটি ক্লাস লিখতে হবে যে ক্লাসটি আসলে এই হিসাব নিকাকেশের কাজগুলো করবে। অনেকটা নিম্নরূপঃ </p>

<pre><code>&lt?php

namespace App\Services;

class Calculator

{

public function add($number1, $number2){

return $number1 + $number2;

}

public function sub($number1, $number2){

return $number1 - $number2;

}

}</code></pre><br>

<p><strong>২।</strong> এবার আমাদের সেই বিখ্যাত Facade ক্লাসটি লিখে ফেলিঃ </p>

<pre><code>&lt?php

namespace App\Facades;

use Illuminate\Support\Facades\Facade;

class Math extends Facade

{

protected static function getFacadeAccessor(){

return 'Math';

}

}</code></pre><br>

<p>হ্যা, এই টুকুই, আর কিছু লিখতে হবেনা এই ক্লাসে। এটাই আসল জাদু… </p><br>

<p><strong>৩। </strong>এইবার আমাদের সার্ভিস প্রোভাইডারে Math স্ট্রিং-এর সাথে ক্লাসকে বাইন্ড করতে হবে।</p>

<pre><code>&lt?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

use App\Services\Calculator;

class AppServiceProvider extends ServiceProvider

{

public function register(){

$this->app->bind('Math', Calculator::class);

}

}</code></pre><br>

<p>এই মানে আমরা AppServiceProvider এ বলতেছি, কেউ যদি Math স্ট্রিং দিয়ে লারাভেলের সার্ভিস কন্টেইনারকে বলে আমাকে একটা ক্লাস দাও তাহলে সে Calculator ক্লাস এর ইন্সটেন্স রিটার্ন করবে।

ব্যাস, আমাদের Facade তৈরি করা হয়ে গেল! এখন আমরা কোন একটি কন্ট্রোলারে নিচের মত কোড লিখে আমাদের নতুন বানানো Facade-টি টেস্ট করতে পারব। </p>

<pre><code>&lt?php

namespace App\Http\Controllers;

use App\Facades\Math;

class TestController extends Controller

{

public function testMath(){

return Math::add(5, 10); //will return 15

}

}</code></pre><br>

<h2>কিভাবে কাজ করে? </h2>

<p>এখন প্রশ্ন হলো উপরে আমরা Facade বানানোর জন্য যে কোড গুলো লিখেছি, সেই কোড গুলো আসলে কিভাবে একটা আরেকটার সাথে সম্পর্কিত? আর এখানে Calculator ক্লাসে কোন স্টেটিক মেথড না থাকা সত্যেও আমরা কিভাবে Math::add(5, 10) কল করতে পারতেছি? </p>

<p>এই প্রশ্ন গুলোর উত্তর খোঁজার জন্য আমরা আমাদের testMath মেথড থেকে শুরু করে যদি পেছন দিকে যায় তাহলে কি দেখতে পাব? </p>

<p>Math::add(5, 10); এর মানে Math Facade এর add নামের স্টেটিক মেথডকে কে কল করা হচ্ছে। এখন আমরা যদি Math Facade -এ যায় তাহলে দেখতে পাব, এর মধ্যে add নামে কোন স্টেটিক মেথড নেই তবে getFacadeAccessor নামে একটি স্টেটিক মেথড আছে যা Math স্ট্রিং রিটার্ন করতেছে। এর মানে এই মেথডকে ক্লাসের কোন স্টেটিক মেথডকে যদি কেউ কল করে তাহলে সে Service Container -এ দেখবে Math স্ট্রিং -এর জন্য কোন ক্লাস বাইন্ড করা আছে কিনা, যদি থাকে তাহলে ঐ ক্লাসের ইন্সটেন্স এর মধ্যে ঐ মেথডটি কল করবে। আর এই পুরো ব্যাপারটি নিয়ন্ত্রন করছে Math ক্লাসের আব্বা Facade ক্লাস। এই জন্য আমরা Math ক্লাসে শুধু এক লাইনের একটা মেথড লিখেছি, আর কিছু করতে হয়নি।</p>

<p>এই ছিল লারাভেল Facade নিয়ে আজকের আলোচনা, এই লম্বা লিখা পড়ার জন্য সবাইকে ধন্যবাদ। কোন প্রশ্ন কিংবা মতামত থাকলে নিঃসংকোচে মন্তব্য করতে পারেন। </p><br>

<h2>আপডেট(০৮-১১-২০১৭)</h2>

<p>আমি আসলে প্রথমে ভেবেছিলাম Http নামে একটা Facade বানিয়ে উদাহরণ দিব, তাই Http নামের Facade এর জন্য কোড লিখেছিলাম। কিন্তু Http দিয়ে উদাহরণ দিলে অনেক বিগিনারদের কাছে জটিল মনে হতে পারে ভেবে পরে Math Facade দিয়ে উদাহরণ দিয়েছি। কিন্তু কিছু কিছু জায়গায় আগের Http কোড রয়ে গিয়েছিল। এই ব্যাপারটা যারা যারা আমাকে জানিয়েছেন তাদের সবাইকে ধন্যবাদ। আমি কোড ঠিক করে দিয়েছি। </p>