**CREATING A NEW LARAVEL APPLICATION – SAMPLE REST API**

Steps:

1. Open the terminal and CD into your desired directory within WAMP server, for instance:

***C:\wamp\www\githubprojects***

1. Type the following command to create new Laravel application:

***Composer create-project Laravel/Laravel projectone***

1. CD into the newly created project in the terminal, for instance:

***C:\wamp\www\githubprojects\githubprojectone***

1. To run the application, type this:

***Php artisan serve***

This will output response like this

Starting Laravel development server: http://127.0.0.1:8000

1. Open the browser and type the above url.
2. You can also access the website normally by typing the web address like this:

http://localhost/githubprojects/githubprojectone/public/

1. To prevent exposing the files in the root directory in WAMP, a little adjustment will be made.

***Rename the server.php as index.php and cut the .htaccess in the public folder and paste it into the root directory. The content of .htaccess looks like this:***

<IfModulemod\_rewrite.c>

<IfModulemod\_negotiation.c>

Options -MultiViews -Indexes

</IfModule>

RewriteEngine On

# Handle Authorization Header

RewriteCond %{HTTP:Authorization} .

RewriteRule .\* - [E=HTTP\_AUTHORIZATION:%{HTTP:Authorization}]

# Redirect Trailing Slashes If Not A Folder...

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteCond %{REQUEST\_URI} (.+)/$

RewriteRule ^ %1 [L,R=301]

# Send Requests To Front Controller...

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteRule ^ index.php [L]

</IfModule>

***While the content of server.php renamed as index.php looks like :***

/\*\*

\* Laravel - A PHP Framework For Web Artisans

\*

\* @package Laravel

\* @author Taylor Otwell<taylor@laravel.com>

\*/

$uri = urldecode(

parse\_url($\_SERVER['REQUEST\_URI'], PHP\_URL\_PATH) ?? ''

);

// This file allows us to emulate Apache's "mod\_rewrite" functionality from the

// built-in PHP web server. This provides a convenient way to test a Laravel

// application without having installed a "real" web server software here.

if ($uri !== '/' &&file\_exists(\_\_DIR\_\_.'/public'.$uri)) {

return false;

}

require\_once \_\_DIR\_\_.'/public/index.php';

1. Create a **controller** to receipt the post from flutter or external endpoint like this:

***Php artisan make:controllerMyApiReceiver***

1. You will see the newly created MyApiReceiver.php file in this path:

***C:\wamp\www\githubprojects\githubprojectone\app\Http\Controllers\MyApiReceiver.php***

1. Type the method the receive the parameters from api , it should look like this:

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class MyApiReceiver extends Controller

{

public function testing(Request $request)

{

$validator= Validator::make($request->all(),[

'memberid' => 'required',

'name' => 'required ',

'amount' => 'required',

]);

if($validator->fails()){

return response()->json([

'status' => 'error',

'message' => 'Validation Fails',

], 401);

}

$memberid = $request->input('memberid');

$name = $request->input('name');

$amount = $request->input('amount');

$result=array('memberid'=>$memberid,'name'=>$name,'amount'=>$amount);

return response()->json([

'status' => 'success',

'message' => $result,

], 401);

}

}

1. Create the route for the controller, open this file:

C:\wamp64\www\githubprojects\githubprojectone\routes\api.php

It should look like this:

***use App\Http\Controllers\MyreceiverController;***

***Route::post('/testing', [MyreceiverController::class, 'testing']);***

***Note, remember to add this to the top of the api.php :  
use App\Http\Controllers\MyreceiverController;***

1. We can now test by posting into it from flutter app or postman.

POSTMAN SCREEN SHOT:

