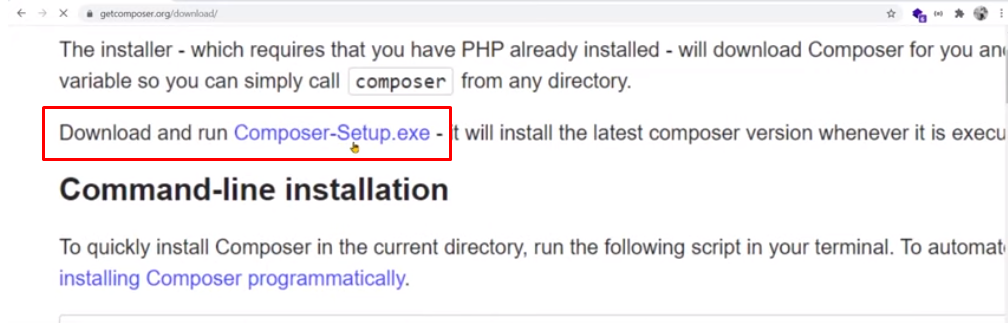
**LARAVEL DOCUMENTAION**

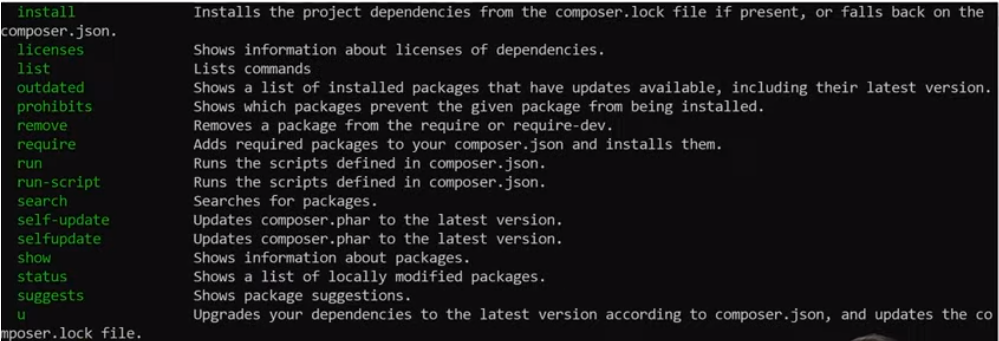
1. **For laravel we need to used cmd. For laravel firstly we need to install the composer.For that use**

**url : <https://getcomposer.org/>**



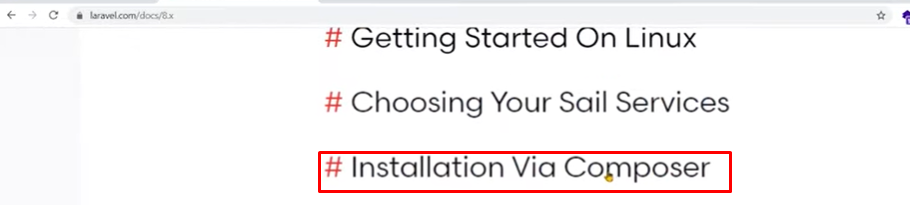
**Aftre download we need to install in our system and confirm composer is properly install or not. For that we need use below path on out cmd:**

**Command : C:\Users\projectname\composer**



**After install composer we need to install laravel framework. For that we need to use laravel.com site.**

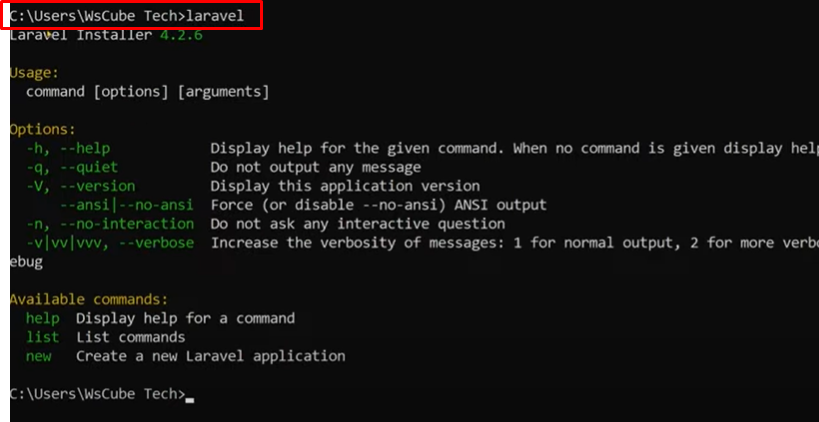
**Click on belolw :**



**After below command use in cmd and install framework :**



**After install framework need to confirm framework is properly install or not. For that in cmd use below command :**



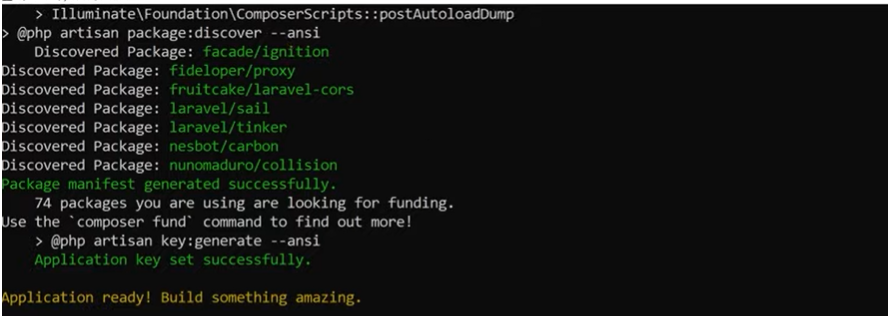
**Our framework is properly install.**

1. **Now we need to create one project, for that we need to navigate our folder in cmd. For that in cmd our path is as belows:**

**C://xampp/htdoc/projectfolder.**

**D:\xampp\htdocs>laravel new foldername**

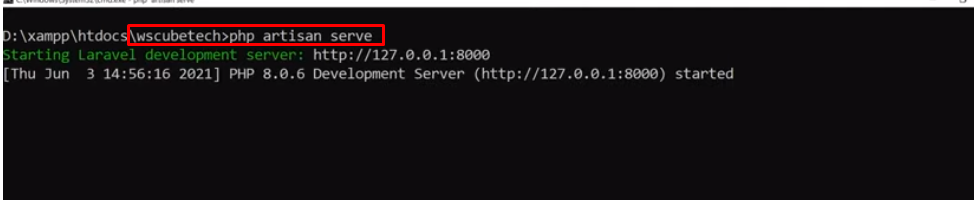




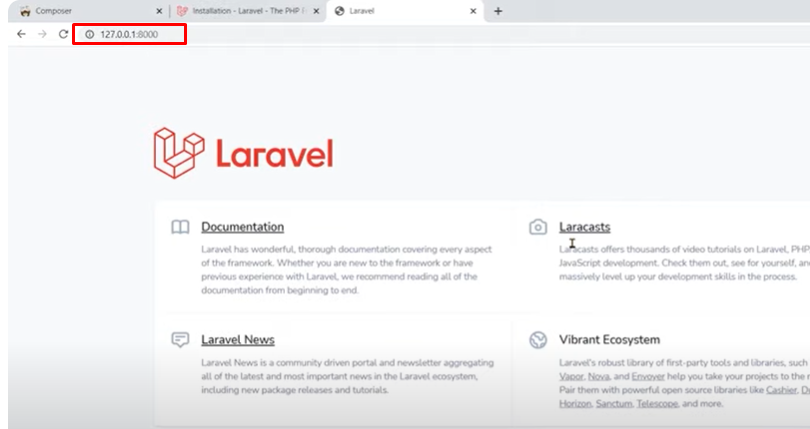
**Laravel framecode setup is successfully downloaded.**

**For laravel we use the blade template.  
Now we need to start laravel for that use below command :**

**Cmd : php artisan serve**



**Use that port in browser now our setup is done.**



1. **Basic structure of routes :**

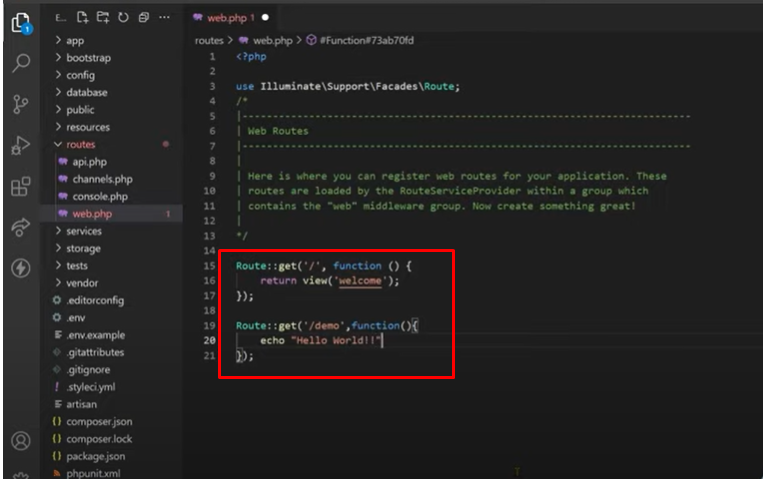
**For setup the routes we need to add routes in Routes folder. In web.php file.**

**There we add our custom routes path. In routes we directly call the view file.**

**We save the file with always filename.blade.php extension.**

**In routes we use the get,post,put,delete all methods. There all method syntax is same.**

**There syntax as below:**



1. **Basic structure of view :**

**In view we merge the html code and php code also. There not need to define the always php sytax there.**

**In view when display the data , there we always use the {{ }} this brackets.**

**Ex. <h1> {{$name}} </h1>**

**This $name is comes from the routes and there value shown in view.**

**For display data we use two types of syntax:**

**- {{ $name }} -> This is not decode the html entity.**

**- {!! $name !!} -> This is decode the html entity.**

1. **Basic structure of controller:**

**In controller there are three basic types:**

**- Basic controller -> accordingly define for our use. There only give single class**

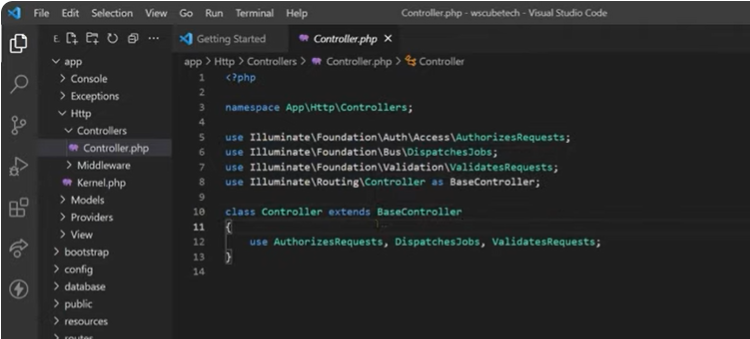
**- Single action controller -> handle for the single request**

**- Resource controller -> CURD oprations. There all functionality is in built.**

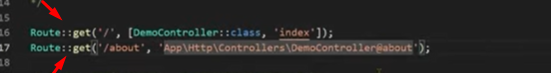
**For create controller se below cmd :**

**Php artisan make:controller ControllerName -> Basic controller**





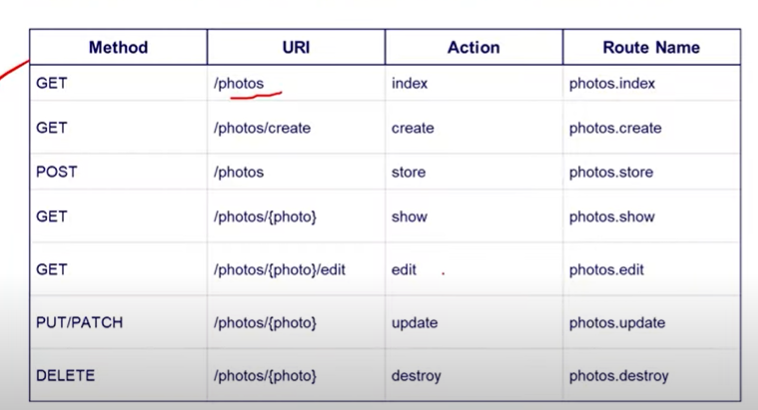
**In route we handle the controller as below ways:**



**Php artisan make:controller ControllerName -invokable ->Single action controller**

**Php artisan make:controller ControllerName -resource ->Resource controller**

**Action handle by resource controller**



1. **Create new component :**

**Component is used for the avoid the code duplications. In view file we create input type many times so using component we avoid that.**

**Create new component use following command:**

**Php artisan make:component ComponentName**

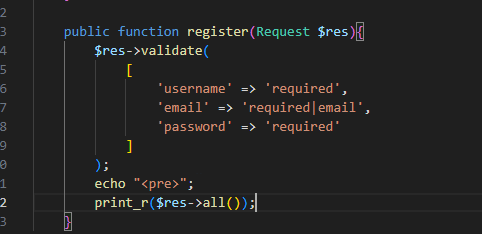
**After creating component there are many files and folder are created automatically.**

**- app -> view/component**

**- view -> component**

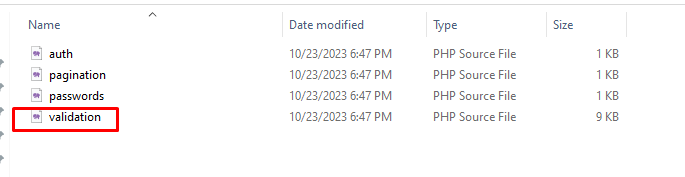
1. **Validations:**

**In laravel there are inbuilt validations are available. For using that validation need to use $res -> validate();**



**There you can handle validations, also create your own validation in that file.**

**File path : \resources\lang\en**



1. **Database configurations:**

**- Create databse in phpmyadmin.**

**- When create database that automatically render in .env file.**

**- Again configure our created database using below command:**

**Php artisan config:Cache**

**- After configuration we used below command for the migrations :**

**Php artisan migrate**

**- After migrate command used we check the our database again, there some table are inserted automatically.**

1. **Create new table using migrations command:**

**Php artisan make:migration create\_tablename\_table.**

**- After using this command there one file generate in the migration folder.**

**- That file name is save using table name.**

**- Open that file include our column there and again migrate command use. Now our table added into DB.**

**- If we add column in existing table then use below command:**

**Php artisan make:migration add\_columns\_to\_tablename\_table.**

**- After add column again use migrate command.**

1. **Create model:**

**- Model create using below command:**

**Php artisan make:model ModelName**

1. **Loops:**

**When we apply loop in our functionality it always start with the @. We use all type of loops in laravel.**

**- IF :  
 @if(codn)**

**{{$name}}**

**@endif**

**- elseIF:**

**@if(codn)**

**Echo “codn mathc”;**

**@elseif(codn)**

**Echo “codn in elseif”;**

**@else**

**Echo “codn in else”;**

**@endif**

# **Accessor and Mutator :**

**- When we store data in table and there are come criteria means username is stored in capital letter that time we use this functionality.**

**Write functions for that in model.like as below**

**- In mutator we use set. When we inserting data that time we have modify our data we use set.**

**Public function setNameAttribute($value)**

**{**

**$this->attributes[‘name’] = ucwords($value);**

**}**

**- In Accessor we use get. When we fetch data and fetching time we have modify our data we use get.**

1. **Session:**

**- There are two type of session. Get and Set. THis define in route**

**- get all session information use below:**

**Route::get(‘get-all-session’,function ()**

**({**

**$session = session->all();**

**})**

**- set session**

**Route::get(‘set-session’,function(Request, $request){**

**$request->session()->put(‘username’,’poonam’);**

**})**

**- destroy session**

**Route::set(‘destory-session’,function(){**

**Session()->forget([‘user\_name’]);**

**})**

1. **Seeder and Faker:**

**- Seeder is use to insert multiple data in table.**

**- Creater seeder using below command:**

**Php artisan make:seeder SeederCustomer**

**For using seeder we already have table for inserting data in that tbl.**

**- After adding info in seeder use below command:**

**Php artisan db:seed**

**- Faker is use to create different name of data randomly.**

**- After adding faker values use below command:**

**Php artisan db:seed**

**-Whenever you execute db command automatically insert data into db.**

1. **Route Grouping:  
   - When we use same keyword in url that time create route group and in that add that variable. Like as below:  
   - If we use customer/create, customer/trash, customer/edit**

**- Here customer keyword is use always so we create group for that.**

**- Route::group([‘prefix’->‘/customer’], function(){**

**-- here put your all get function which is use /customer**

**})**

1. **Localization:  
   - If we need data in multiple language that time use localization.**

**- For that create lang folder as per req and add in web.php structas below:  
Route::get(‘/{lang}?’), funation($lang == null){**

**App :: setLocal($lang);**

**Return view(‘’);**

**}**

1. **Stub:**

**- It is used for control the all artisan command which is executed automatically. Like model,controller**

**- When we create controller that time there we need index function by default, so we customize stub structure.**

**- For display stub use below command,**

**Php artisan stub:publish**

**-After use this command stub folder is shown on root.**

1. **Middleware :**

**- It is basically use for the filter the request.**

**- It filter all request which is go to the application. That request is harmfull or not for application is decide through the middleware.**

**- Use below command to create middleware**

**Php artisan make:middleware Abc**

**- After create middleware add our file in kernal.php file.**

**- Two type - Route middleware and Group middleware.**