**What is Laravel 9.0 ?**

Laravel is a framework of php.

Laravel is most popular framework in php

Laravel provides its own structure i.e support MVC architecture.

Laravel is support oops.

Laravel is best framework to create dynamic website or web application.

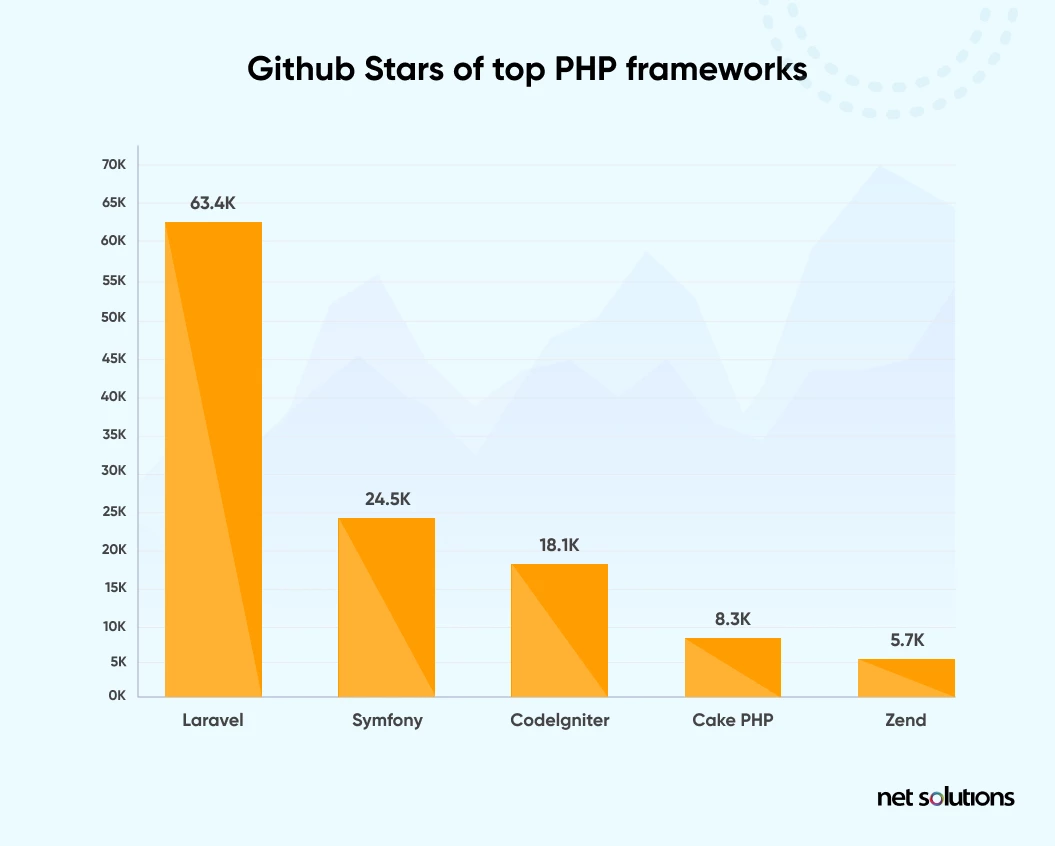
Laravel is most used to create web apps | website | mobile apps api’s.

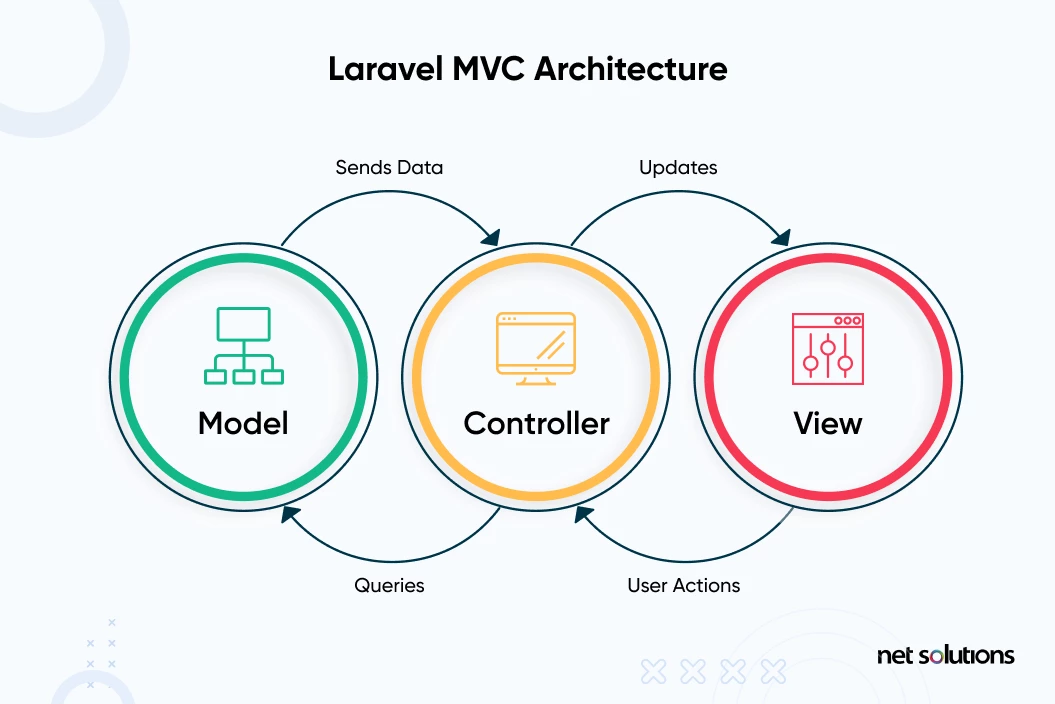
Laravel provides seo friendly website.

Laravel best for large or small scale of website or web apps.

**Advantage of Laravel**

1. Framework of php
2. Open source framework
3. Seo friendly provides website
4. Mvc support
5. Blade templating structure
6. Database migrations procedures
7. Large scale developed website
8. Secured website development
9. Laravel is used for frontend and backend





**Laravel installations:**

**Step 1: download and install composer**

**What is composer ?**

<https://getcomposer.org/download/>

composer is an dependency manager i.e used to install Laravel .

Composer in Laravel is used **to manage their dependencies**. Once Composer is installed, download the framework and extract its contents into a directory on your server. The primary task you should do after installing Laravel is to set your application key to a random string

How to check composer is install or not ?

Step 2: check composer install or not ?

Cmd: composer

C:\xampp\htdocs>composer

\_\_\_\_\_\_

/ \_\_\_\_/\_\_\_ \_\_\_\_ \_\_\_ \_\_\_\_ \_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_

/ / / \_\_ \/ \_\_ `\_\_ \/ \_\_ \/ \_\_ \/ \_\_\_/ \_ \/ \_\_\_/

/ /\_\_\_/ /\_/ / / / / / / /\_/ / /\_/ (\_\_ ) \_\_/ /

\\_\_\_\_/\\_\_\_\_/\_/ /\_/ /\_/ .\_\_\_/\\_\_\_\_/\_\_\_\_/\\_\_\_/\_/

/\_/

Composer version 2.4.1 2022-08-20 11:44:50

**Step 3: install Laravel**

[**https://laravel.com/docs/9.x**](https://laravel.com/docs/9.x)

However, if you are not using Homestead, you will need to make sure your server meets the following requirements:

* PHP >= 7.2.5
* BCMath PHP Extension
* Ctype PHP Extension
* Fileinfo PHP extension
* JSON PHP Extension
* Mbstring PHP Extension
* OpenSSL PHP Extension
* PDO PHP Extension
* Tokenizer PHP Extension
* XML PHP Extension

composer create-project laravel/laravel example-app

Step 5: how to run projects

C:\xampp\htdocs\charmi-fashionshop-app>

C:\xampp\htdocs\charmi-fashionshop-app>php artisan serve

INFO Server running on [http://127.0.0.1:8000].

Press Ctrl+C to stop the server

Server running url of Laravel

<http://127.0.0.1:8000/>

**Laravel provides maintenance mode :**

C:\xampp\htdocs\charmi-fashionshop-app>php artisan down

INFO Application is now in maintenance mode.

C:\xampp\htdocs\charmi-fashionshop-app>

C:\xampp\htdocs\charmi-fashionshop-app>php artisan up

INFO Application is now in maintenance mode.

C:\xampp\htdocs\charmi-fashionshop-app>

**How to run Laravel on localhost :**

Public/index.php => paste in main directory

Change path of n

require \_\_DIR\_\_.'../vendor/autoload.php';

$app = require\_once \_\_DIR\_\_.'../bootstrap/app.php';

<http://localhost/flipkart-app/>

**how to run Laravel on server development URL**

**cmd :** php artisan serve

<http://127.0.0.1:8000/>

**how to connect Laravel with database**

go in main director and change .env

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=flipkart\_db

DB\_USERNAME=root

DB\_PASSWORD=

How to connect Laravel more than one database used

Config=>database.php

'mysql' => [

            'driver' => 'mysql',

            'url' => env('DATABASE\_URL'),

            'host' => env('DB\_HOST', '127.0.0.1'),

            'port' => env('DB\_PORT', '3306'),

            'database' => env('DB\_DATABASE', 'flipkart\_db'),

            'username' => env('DB\_USERNAME', 'root'),

            'password' => env('DB\_PASSWORD', ''),

            'unix\_socket' => env('DB\_SOCKET', ''),

            'charset' => 'utf8mb4',

            'collation' => 'utf8mb4\_unicode\_ci',

            'prefix' => '',

Laravel migrations:

Laravel migrations is a default mechanism to create a database tables structure or its schema.

How to create a tables in Laravel using migrations

Cmd : php artisan make:migration create\_customers\_table

How to migrate tables inside of database :

Cmd : php artisan migrate

php artisan list

php artisan migrate

migrate:fresh Drop all tables and re-run all migrations

migrate:install Create the migration repository

migrate:refresh Reset and re-run all migrations

migrate:reset Rollback all database migrations

migrate:rollback Rollback the last database migration

migrate:status Show the status of each migration

**Laravel MVC**

**Model : database connection and create a member function**

**View :view create a UI**

**Controller: controller handle a logic parts**

MVC Architecture Of Laravel

MVC is **an architectural design pattern that helps to develop web applications faster**. MVC stands for Model-View-Controller



**Service Provider :**

Service providers are the central place of all Laravel application bootstrapping. Your own application, as well as all of Laravel's core services, are bootstrapped via service providers.

But, what do we mean by "bootstrapped"? In general, we mean **registering** things, including registering service container bindings, event listeners, middleware, and even routes. Service providers are the central place to configure your application.

If you open the **config/app.php** file included with Laravel, you will see a providers array. These are all of the service provider classes that will be loaded for your application. By default, a set of Laravel core service providers are listed in this array. These providers bootstrap the core Laravel components, such as the mailer, queue, cache, and others. Many of these providers are "deferred" providers, meaning they will not be loaded on every request, but only when the services they provide are actually needed.

**config/app.php**

**Model =>app\Models=>User.php**

<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable

{

    use HasApiTokens, HasFactory, Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array<int, string>

     \*/

    protected $fillable = [

        'name',

        'email',

        'password',

    ];

    /\*\*

     \* The attributes that should be hidden for serialization.

     \*

     \* @var array<int, string>

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

    /\*\*

     \* The attributes that should be cast.

     \*

     \* @var array<string, string>

     \*/

    protected $casts = [

        'email\_verified\_at' => 'datetime',

    ];

}

**How to create a model**

cmd : php artisan make:model Customer

**how to create a controller**

cmd : php artisan make:controller CustomerController

cmd : php artisan make:controller CustomerController --resource

**how to create view**

**resources\views**

name.blade.php

**how to create routing**

**Routing :**

The most basic Laravel routes accept a URI and a closure, providing a very simple and expressive method of defining routes and behavior without complicated routing configuration files:

use Illuminate\Support\Facades\Route;

Route::get('/greeting', function () {

return 'Hello World';

});

#### [The Default Route Files](https://laravel.com/docs/9.x/routing#the-default-route-files)

All Laravel routes are defined in your route files, which are located in the routes directory. These files are automatically loaded by your application's App\Providers\RouteServiceProvider. The routes/web.php file defines routes that are for your web interface. These routes are assigned the web middleware group, which provides features like session state and CSRF protection. The routes in routes/api.php are stateless and are assigned the api middleware group.

use App\Http\Controllers\UserController;

Route::get('/user', [UserController::class, 'index']);

**How to create Login | register | session | logout in Laravel 9.0 using Authentication and Laravel Breeze**

[Laravel Breeze](https://github.com/laravel/breeze) is a minimal, simple implementation of all of Laravel's [authentication features](https://laravel.com/docs/9.x/authentication), including login, registration, password reset, email verification, and password confirmation. Laravel Breeze's default view layer is made up of simple [Blade templates](https://laravel.com/docs/9.x/blade) styled with [Tailwind CSS](https://tailwindcss.com). Or, Breeze can scaffold your application using Vue or React and [Inertia](https://inertiajs.com).

### [Installation](https://laravel.com/docs/9.x/starter-kits#laravel-breeze-installation)

First, you should [create a new Laravel application](https://laravel.com/docs/9.x/installation), configure your database, and run your [database migrations](https://laravel.com/docs/9.x/migrations). Once you have created a new Laravel application, you may install Laravel Breeze using Composer:

composer require laravel/breeze --dev

### [Breeze & Blade](https://laravel.com/docs/9.x/starter-kits#breeze-and-blade)

After Composer has installed the Laravel Breeze package, you may run the breeze:install Artisan command. This command publishes the authentication views, routes, controllers, and other resources to your application. Laravel Breeze publishes all of its code to your application so that you have full control and visibility over its features and implementation.

The default Breeze "stack" is the Blade stack, which utilizes simple [Blade templates](https://laravel.com/docs/9.x/blade) to render your application's frontend. The Blade stack may be installed by invoking the breeze:install command with no other additional arguments. After Breeze's scaffolding is installed, you should also compile your application's frontend assets:

php artisan breeze:install

php artisan migrate

npm install

npm run dev