wemwgjøvwni ivngvwbi ivwng

Follow on: https://www.youtube.com/watch?v=L6xaLfuqGfU&list=PLjsp2uDzfb31NqnycsDO41j18bYeKOGp-&index=4&ab\_channel=WebJourney  
jviv‡fj ïiæ:

1. composer create-project --prefer-dist laravel/laravel blog

Zvici

1. php artisan serve

KgvÛ w`‡q †`Le mvfv©i ivb K‡i wKbv Ges[**http://127.0.0.1:8000/**](http://127.0.0.1:8000/) GLv‡b wM‡q †`Le mvBU wVKg‡Zv BÝUªj n‡q‡Q wKbv | wVK \_vK‡j mvBU †`Lv‡e |  
Gevi mivmwi P‡j hv‡ev ‡KvW GwWU‡i Ges cª‡Rw± I‡cb Kie |

Gici .env dvBj I‡cb K‡i WvUv‡e‡mi bvg BDRvi bvg I cvmIqvW© ewm‡q †mf w`‡ev |

1. DB\_CONNECTION=mysql
2. DB\_HOST=127.0.0.1
3. DB\_PORT=3306
4. DB\_DATABASE=laravel
5. DB\_USERNAME=root
6. DB\_PASSWORD= 123

Gevi GKUv K‡›Uªvjvi evbv‡bvi Rb¨ wb‡Pi KgvÛ w`‡ev

1. php artisan make:controller ProductController

K‡›Uªvjvi ˆZwi nIqvi ci Avgiv K‡›Uªvjvi I‡cb K‡i Gfv‡e wjLe

1. <?php
3. namespace App\Http\Controllers;
4. **use** Illuminate\Http\Request;
6. **class** ProductController **extends** Controller
7. {
8. **public** **function** index(){
9. **return** 'All Product';
10. }
11. }

**Gevi P‡j hv‡ev web.php dvB‡j Ges ivDU wjLe | cÖ‡Z¨KwU ivD‡Ui Rb¨ Dc‡i K‡›Uªvjvi bvg cv\_ e¨envi Ki‡Z n‡e |**

1. **use** App\Http\Controllers\ProductController;
3. Route::resource('products', ProductController::**class**);
4. **A\_ev Gfv‡e**
5. Route::get('/products', [ProductController::**class,** 'index']);

**Gevi http://127.0.0.1:8000/products GB G‡Wª‡m wM‡q †`Le wVKg‡Zv KvR K‡i wKbv |**All Product **wjLvwU Avm‡e †c‡R |**

**Gevi hw` BDAviGj G c¨vivwgUvi cvm Ki‡Z PvB Zvn‡j Gfv‡e wjL‡Z n‡e |**

1. Route::get('/products/{name}', [ProductController::**class,** 'index']);

**Gevi GB c¨vivwgUviwU‡K K‡›Uªvjv‡i wiwmf Ki‡Z n‡e |**

1. <?php
3. namespace App\Http\Controllers;
4. **use** Illuminate\Http\Request;
6. **class** ProductController **extends** Controller
7. {
8. **public** **function** index($name = **null**){
9. **return** 'All Product '.$name;
10. }
11. }

**Gevi http://127.0.0.1:8000/products/anything GB G‡Wª‡m wM‡q †`Le wVKg‡Zv KvR K‡i wKbv |**All Product anything **wjLvwU Avm‡e †c‡R |**

**GLv‡b hw` c¨vivwgUvi Lvwj \_v‡K Zvn‡j Gii Avm‡e ZvB GB Gii hv‡Z bv Av‡m Gi Rb¨ c¨vivwgUvi wjLe Gfv‡e**

Route::get('/products/{name**?**}', [ProductController::**class,** 'index']);   
**GLv‡b GKUv ? wPý e¨envi Ki‡Z n‡e Ges K‡›Uªvjv‡i $name=null wjL‡jB n‡e Avi Gii †`Lv‡e bv c¨vivwgUvi \_vKzK Avi bv \_vKzK | ivD‡Ui hw` bvg w`‡q Pvq Zvn‡j ivD‡Ui c‡i Gfv‡e bvg wjL‡Z nq |**

1. Route::get('/products/{name}', [ProductController::**class,** 'index'])->name('allproduct');

|1. <a href="{{ route('allproduct ')}}">Link</a>

**Gevi wfD dvBj wb‡q KvR Kie | cÖ\_‡g jviv‡fj WvB‡i±wi¸‡jvi ga¨ †\_‡K wi‡mv‡P©m WvB‡i±wi‡Z hv‡ev| Gevi wfDm WvD‡i±wi‡Z hv‡ev | GLv‡b wM‡q †`Lb welcome.blade.php bvgK GKUv dvBj Av‡Q | †mUv PvB‡jB GwWU Ki‡Z cvwi A\_ev Avgiv bZzb GKv wfD dvBj ˆZwi K‡i KvR Kie | Pjyb †`Lv hvK**

**‡hgb user.blade.php GLv‡b GKUz †KvW wjwL | <h1>User Page</h1> Gevi †mf K‡i P‡j hvB ivDU dvB‡j †hLv‡b Avgiv bZzb GKUv ivDU ˆZwi Kie 3 wbq‡g | wZbfv‡eB wjLv hvq|**

1. Route::get('/user', function(){
2. return view('user'); // This user is blade file name
3. });
4. // Method 2 Like This
5. Route::view('/user','user'); // 1st user url Name and last user blade file name. No way to Pass perameter
6. Route::get('/user3', [ProductController::**class**, 'getuser3']);
7. // go to controller and write this method
8. **public** **function** getuser3(){
9. **return** view('user');
10. }

**Gevi Avwm K‡›Uªvjvi †\_‡K WvUv cvm Kivi wbq‡g Kg‡c± e¨envi K‡i |**

1. <?php
3. namespace App\Http\Controllers;
5. **use** Illuminate\Http\Request;
7. **class** UserController **extends** Controller
8. {
9. **public** **function** index(){
10. //$welcome = 'Welcome to Iqbal world';
11. $welcome2 = array(
12. ‘name’ => ‘Iqbal’,
13. ‘age’ => ‘33’,
14. ‘hobby’ => ‘Palying’
15. );
16. **return** view('user', compact('welcome2'));
17. }
18. }

Gevi BDRvi ‡eøW †c‡R wM‡q WvUv wiwmf Kie Gfv‡e   
**{{ $welcome }} or {{$welcome[‘name’]}} {{$welcome[‘age’]}}**

**Route wjLvi me iKg wbqg:**

1. Route::get('/greeting', **function** () {
2. **return** 'Hello World';
3. });

// Method 2

1. **use** App\Http\Controllers\UserController;
2. Route::get('/user', [UserController::**class**, 'index']);

// Method 3 

1. Route::get('/user', [App\Http\Controllers\UserController::**class**, 'index']);

Available Router Methods

1. Route::**get**($uri, $callback);
2. Route::**post**($uri, $callback);
3. Route::**put**($uri, $callback);
4. Route::**patch**($uri, $callback);
5. Route::**delete**($uri, $callback);
6. Route::**options**($uri, $callback);

// Method 3

1. Route::match(['get', 'post'], '/', **function** () {
2. //
3. });
5. Route::any('/', **function** () {
6. //
7. });

// Method 4

1. **use** Illuminate\Http\Request;
2. Route::get('/users', **function** (Request $request) {
3. //
4. });
5. Route::redirect('/here', '/there');
6. Route::redirect('/here', '/there', 301);
7. Route::permanentRedirect('/here', '/there');
8. Route::controller('home', 'HomeController');
9. Route::group(['middleware' => 'auth'], **function** () {
10. Route::get('/', **function** ()    {
11. // Uses Auth Middleware
12. });
13. });
14. Route::group(['namespace' => 'Admin'], **function**()
15. {
16. // Controllers Within The "App\Http\Controllers\Admin" Namespace
17. Route::group(['namespace' => 'User'], **function**() {
18. // Controllers Within The "App\Http\Controllers\Admin\User" Namespace
19. });
20. });
21. Route::group(['domain' => '{account}.myapp.com'], **function** () {
22. Route::get('user/{id}', **function** ($account, $id) {
23. //
24. });
25. });

1. Route::group(['prefix' => 'admin'], **function** () {
2. Route::get('users', **function** ()    {
3. // Matches The "/admin/users" URL
4. });
5. });
6. Route::group(['prefix' => 'accounts/{account\_id}'], **function** () {
7. Route::get('detail', **function** ($accountId)    {
8. // Matches The "/accounts/{account\_id}/detail" URL
9. });
10. });

//Method 5

1. Route::view('/welcome', 'welcome');
2. Route::view('/welcome', 'welcome', ['name' => 'Taylor'])

// Method 6

1. Route::get('/user/{id}', **function** ($id) {
2. **return** 'User '.$id;
3. });
4. Route::get('/posts/{post}/comments/{comment}', **function** ($postId, $commentId) {
5. **return** 'User '.$id;
6. });

// Method 7

1. **use** Illuminate\Http\Request;
3. Route::get('/user/{id}', **function** (Request $request, $id) {
4. **return** 'User '.$id;
5. });
6. Route::get('/user/{name?}', **function** ($name = null) {
7. **return** $name;
8. });
9. Route::get('/user/{name?}', **function** ($name = 'John') {
10. **return** $name;
11. });

// Route Method 8

1. Route::get('/user/{name}', **function** ($name) {
2. //
3. })->where('name', '[A-Za-z]+');
5. Route::get('/user/{id}', **function** ($id) {
6. //
7. })->where('id', '[0-9]+');
9. Route::get('/user/{id}/{name}', **function** ($id, $name) {
10. //
11. })->where(['id' => '[0-9]+', 'name' => '[a-z]+']);

// Method 9

1. Route::get('/user/{id}/{name}', **function** ($id, $name) {
2. //
3. })->whereNumber('id')->whereAlpha('name');
5. Route::get('/user/{name}', **function** ($name) {
6. //
7. })->whereAlphaNumeric('name');
9. Route::get('/user/{id}', **function** ($id) {
10. //
11. })->whereUuid('id');

// Method 8

1. Go - App\Providers\RouteServiceProvider
3. **public** **function** boot()
4. {
5. Route::pattern('id', '[0-9]+');
6. }
7. Route::get('/user/{id}', **function** ($id) {
8. // Only executed if {id} is numeric...
9. });

// Method 10

1. Route::get('/search/{search}', **function** ($search) {
2. **return** $search;
3. })->where('search', '.\*');

// Method 11

1. Route::get('/user/profile', **function** () {
2. //
3. })->name('profile');
4. Route::get('/user/profile',[UserProfileController::**class**, 'show'])->name('profile');
5. // Generating URLs...
6. {{ route('profile') }}
7. // Generating Redirects...  
   **return** redirect()->route('profile');
8. Route::get('/user/{id}/profile', **function** ($id) {
9. //
10. })->name('profile');
11. $url = route('profile', ['id' => 1]);
12. Route::get('/user/{id}/profile', **function** ($id) {
13. //
14. })->name('profile');
15. $url = route('profile', ['id' => 1, 'photos' => 'yes']);
16. // /user/1/profile?photos=yes
17. /\*\*
18. \* Handle an incoming request.
19. \*
20. \* @param  \Illuminate\Http\Request  $request
21. \* @param  \Closure  $next
22. \* @return mixed
23. \*/
24. **public** **function** handle($request, Closure $next)
25. {
26. **if** ($request->route()->named('profile')) {
27. //
28. }
29. **return** $next($request);
30. }

// Route Method 11

1. Route::middleware(['first', 'second'])->group(**function** () {
2. Route::get('/', **function** () {
3. // Uses first & second middleware...
4. });
6. Route::get('/user/profile', **function** () {
7. // Uses first & second middleware...
8. });
9. });
10. Route::domain('{account}.example.com')->group(**function** () {
11. Route::get('user/{id}', **function** ($account, $id) {
12. //
13. });
14. });
15. Route::prefix('admin')->group(**function** () {
16. Route::get('/users', **function** () {
17. // Matches The "/admin/users" URL
18. });
19. });
20. Route::name('admin.')->group(**function** () {
21. Route::get('/users', **function** () {
22. // Route assigned name "admin.users"...
23. })->name('users');
24. });
25. Route::resource('home', 'HomeController::class');
26. Route::apiResource('photos', PhotoController::**class**);
27. Route::apiResources([
28. 'photos' => PhotoController::**class**,
29. 'posts' => PostController::**class**,
30. ]);
31. **use** App\Http\Controllers\PhotoCommentController;
33. Route::resource('photos.comments', PhotoCommentController::**class**);
34. **// This route will register a nested resource that may be accessed with URIs like the following:**
35. /photos/{photo}/comments/{comment}
36. **use** App\Http\Controllers\PhotoController;
38. Route::resource('photos', PhotoController::**class**)->only([
39. 'index', 'show'
40. ]);
42. Route::resource('photos', PhotoController::**class**)->except([
43. 'create', 'store', 'update', 'destroy'
44. ]);
45. // Check if route is \*\*\*
46. **Request::route()->named("YourRouteNameView")**
47. **{{ Request::path() }}**
48. **use** App\Models\User;
49. Route::get('/users/{user}', **function** (User $user) {
50. **return** $user->email;
51. });
52. **use** App\Http\Controllers\UserController;
53. **use** App\Models\User;
55. // Route definition...
56. Route::get('/users/{user}', [UserController::**class**, 'show']);
57. // Controller method definition...
58. **public** **function** show(User $user)
59. {
60. **return** view('user.profile', ['user' => $user]);
61. }
62. // Soft Delete
63. **use** App\Models\User;
64. Route::get('/users/{user}', **function** (User $user) {
65. **return** $user->email;
66. })->withTrashed();
67. **use** App\Models\Post;
69. Route::get('/posts/{post:slug}', **function** (Post $post) {
70. **return** $post;
71. });
72. //If you would like model binding to always **use** a database column other than id when retrieving a given model **class**, you may override the getRouteKeyName method on the Eloquent model:
74. /\*\*
75. \* Get the route key for the model.
76. \*
77. \* @return string
78. \*/
79. **public** **function** getRouteKeyName()
80. {
81. **return** 'slug';
82. }
83. **use** App\Models\Post;
84. **use** App\Models\User;
86. Route::get('/users/{user}/posts/{post:slug}', **function** (User $user, Post $post) {
87. **return** $post;
88. });
89. **use** App\Models\Post;
90. **use** App\Models\User;
92. Route::get('/users/{user}/posts/{post}', **function** (User $user, Post $post) {
93. **return** $post;
94. })->scopeBindings();
95. Route::scopeBindings()->group(**function** () {
96. Route::get('/users/{user}/posts/{post}', **function** (User $user, Post $post) {
97. **return** $post;
98. });
99. });
100. **use** App\Http\Controllers\LocationsController;
101. **use** Illuminate\Http\Request;
102. **use** Illuminate\Support\Facades\Redirect;
104. Route::get('/locations/{location:slug}', [LocationsController::**class**, 'show'])
105. ->name('locations.view')
106. ->missing(**function** (Request $request) {
107. **return** Redirect::route('locations.index');
108. });
109. **use** App\Models\User;
110. **use** Illuminate\Support\Facades\Route;
112. /\*\*
113. \* Define your route model bindings, pattern filters, etc.
114. \*
115. \* @return void
116. \*/
117. **public** **function** boot()
118. {
119. Route::model('user', User::**class**);
121. // ...
122. }
123. // Next, define a route that contains a {user} parameter:
124. **use** App\Models\User;
126. Route::get('/users/{user}', **function** (User $user) {
127. //
128. });
129. **use** App\Models\User;
130. **use** Illuminate\Support\Facades\Route;
132. /\*\*
133. \* Define your route model bindings, pattern filters, etc.
134. \*
135. \* @return void
136. \*/
137. **public** **function** boot()
138. {
139. Route::bind('user', **function** ($value) {
140. **return** User::where('name', $value)->firstOrFail();
141. });
142. // ...
143. }
144. Route::fallback(**function** () {
145. //
146. });
147. $route = Route::current(); // Illuminate\Routing\Route
148. $name = Route::currentRouteName(); // string
149. $action = Route::currentRouteAction(); // string
150. php artisan route:cache
151. php artisan route:clear
152. **$url = url('foo');**
153. # Define not found route
154. Route::fallback('HomeController@notFound')->name('notFound');
156. # Define 404 route without controller
157. Route::fallback(**function** () {
158. **return** view("404");
159. });
160. **use** App\Http\Controllers\CommentController;
161. Route::resource('photos.comments', CommentController::**class**)->shallow();

This route definition will define the following routes:

| **Verb** | **URI** | **Action** | **Route Name** |
| --- | --- | --- | --- |
| GET | /photos/{photo}/comments | index | photos.comments.index |
| GET | /photos/{photo}/comments/create | create | photos.comments.create |
| POST | /photos/{photo}/comments | store | photos.comments.store |
| GET | /comments/{comment} | show | comments.show |
| GET | /comments/{comment}/edit | edit | comments.edit |
| PUT/PATCH | /comments/{comment} | update | comments.update |
| DELETE | /comments/{comment} | destroy | comments.destroy |

**@CSRF**

1. <form action="/example" method="POST">
2. <input type="hidden" name="\_method" value="PUT">
3. <input type="hidden" name="\_token" value="{{ csrf\_token() }}">
4. </form>
5. <form action="/example" method="POST">
6. @method('PUT')
7. @csrf
8. </form>
9. <input type="hidden" name="\_token" value="<?php echo csrf\_token(); ?>">
10. // Vanilla PHP
11. <?php echo csrf\_field(); ?>
13. // Blade Template Syntax
14. {{ csrf\_field() }}
15. <meta name="csrf-token" content="{{ csrf\_token() }}">
16. Route::get('api/users/{user}', **function** (App\User $user) {
17. **return** $user->email;
18. });

**Middlewire**

1. **public** **function** handle($request, Closure $next)
2. {
3. **if** ($request->input('age') < 200)
4. {
5. **return** redirect('home');
6. }
7. **return** $next($request);
8. }
9. Route::get('admin/profile', ['middleware' => 'auth', **function**()
10. {
11. //
12. }]);
13. Route::get('/age', **function** () {})->middleware('CheckAge');
14. # Group routes that has middleware that checks to see **if** user
15. # Is logged on only logged in users are allowed to access this routes
16. Route::middleware(['isLoggedIn'])->group(**function** () {
17. Route::get('users', 'UserController::class, 'show')->name('users');
18. Route::get('users/:id', 'UserController:class, 'get')->name('user\_get');
19. });
21. # Group routes that are **public** only can be accessed without user is logged on
22. Route::middleware(['web'])->group(**function** () {
23. Route::get('login', 'UserController:class, 'login')->name('login');
24. Route::get('logout', 'UserController:class, 'logout')->name('logout');
25. });

**Template Mastering**

cÖ\_‡g Avgiv †jAvDU Kg‡cv‡b›U evbv‡bv wkLe wWdìUv †\_‡K | †hgb   
<x-app-layout> </x-app-layout> GUv wKfv‡e evbv‡Z nq Avmyb †`‡L †bB |

php artisan make:component IqbalLayout // press Enter

GB KgvÛ †`Iqvi ci `yBUv dvBj ˆZwi n‡e `yB RvqMvq | GKUv ˆZwi n‡e App\View\Components G Ges Av‡iKUv ˆZwi n‡e resources\views\components G | Gevi App/View/Components/IqbalLayout.php dvB‡j wM‡q

1. **public** **function** render()
2. {
3. **return** view('layouts.app');  GLv‡b †h dvBj Gi bvg w`‡ev †mUvB Avm‡e |
4. }

Gevi wfD ‡\_‡K †h‡Kvb †eøW dvB‡ji wfZ‡i GUv e¨envi Ki‡Z n‡j Gfv‡e wjL‡Z n‡e |

<x-iqbal-layout> </x-iqbal-layout>

jvBfIq¨vi Kg‡cv‡b‡›U wWdë †jAvDU app/config/liveware.php ‡jv‡Kkb jvBfIq¨vi dvB‡j †mU Kiv \_v‡K | jvBfIq¨vi Kg‡cv‡b›U evbv‡Z GB KgvÛ w`‡Z nq |  
php artisan make:livewire MasterLayout

MasterLayout ‡c‡R Gfv‡e wjL‡j mve †c‡R wM‡q wb‡Pi g‡Zv wiwmf Ki‡Z n‡e |

1. <!-- Page Heading -->
2. @**if** (isset($header22))
3. <header **class**="bg-white shadow">
4. <div **class**="max-w-7xl mx-auto py-6 px-4 sm:px-6 lg:px-8">
5. {{ $header22 }}
6. </div>
7. </header>
8. @**endif**

View Sub Page Receive header Like This

1. <x-slot name="header22">
2. <h2 **class**="font-semibold text-xl text-gray-800 leading-tight">
3. {{ Auth::user()->name }}
4. </h2>
5. </x-slot>
6. <footer>
7. {{ $footer }}
8. </footer>
9. Sub Page
10. <slot name="footer">This is footer Content</slot>

jvBfIq¨vi Kg‡cv‡b›U e¨envi K‡i WvUv †`Lv‡bvi wbqg: jvBf Kg‡cv‡b›U ˆZwi Ki‡Z wb‡Pi K‡gÛ w`‡Z n‡e Ges wb‡Pi g‡Zv K‡i wjL‡Z n‡e |  
php artisan make:livewire PostComponent

GB KgvÛ Pvjv‡bvi ci 2 RvqMvq 2 Uv dvBj ˆZwi n‡e cÖ\_gUv app\Http\Livewire GLv‡b Avi 2q wU ˆZwi n‡e resources\views\livewire GLv‡b | Gevi 1g dvB‡j wb‡Pi g‡Zv K‡i K‡›Uªvjv‡ii g‡Zv KvR Kie | Avi 2q dvB‡j wfD dvB‡ji KvR Kie Ges jvBfIq¨v‡ii Rb¨ ivDU wjL‡Z nq Gfv‡e |

Route::get('/post', ShowPost::class)->name('post');

1. **class** ShowPost **extends** Component
2. {
3. **public** **function** render()
4. {
5. **return** view('livewire.show-posts', [
6. 'users' => User::all(),
7. ]);
8. }
9. }
10. <x-slot name="header">
11. <h2 **class**="font-semibold text-xl text-gray-800 leading-tight">
13. {{ Auth::user()->name }}
15. @**if**(Auth::check() && Auth::user()->user\_type == "1")
16. {{ \_\_('Admin Dashboard') }}
17. @**else**
18. {{ \_\_('User Dashboard') }}
19. @**endif**
21. @can('isAdmin')
22. <div **class**="btn btn-success btn-lg">
23. You have Admin Access
24. </div>
25. @**else**
26. <div **class**="btn btn-info btn-lg">
27. You have User Access
28. </div>
29. @endcan
31. </h2>
32. </x-slot>
34. <div **class**="py-12">
35. <div **class**="max-w-7xl mx-auto sm:px-6 lg:px-8">
36. <div **class**="bg-white overflow-hidden shadow-xl sm:rounded-lg">
37. <h1>Display All User Using Livewire</h1>
38. **@foreach ($users as $user)**
39. **<ul>**
40. **<li>{{ $user->username }}</li>**
41. **</ul>**
42. **@endforeach**
43. <x-jet-welcome />
44. </div>
45. </div>
46. </div>

1. **class** Product **extends** Model
2. {
3. **use** HasFactory;
5. **protected** $fillable = [
6. 'name', 'slug', 'detail'
7. ];
8. }

g‡Wj Gi wfZ‡ii GB wdì¸‡jv mwVKfv‡e \_vK‡Z n‡e | Zvici GB wb‡Pi KgvÛwU Pvjv‡bv Av‡M BDRvi g‡Wj Gi GKUv d¨v±ix evwb‡q wb‡Z n‡e hw` bv \_v‡K | wWdì BDRvi g‡Wj Gi d¨v±ix evbv‡bv \_v‡K | bv nq Avgiv wb‡Pi KgvÛ w`‡q GKUv d¨v±ix evwb‡q wb‡ev |

|php artisan make:factory ProductFactory --model=Product

Ges Gi wfZ‡i wb‡Ri g‡Zv K‡i †KvW wjL‡Z n‡e |

1. **public** **function** definition()
2. {
3. **return** [
4. 'name' => $this->faker->name(),
5. 'username' => $this->faker->username(),
6. 'email' => $this->faker->unique()->safeEmail(),
7. 'email\_verified\_at' => now(),
8. 'password' => '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', // password
9. 'remember\_token' => Str::random(10),
10. ];
11. }

composer dump-autoload // Run This Command before tinker if any error in tinker

php artisan tinker // Press Enter and Type Below Command  
App\Models\User::factory()->count(20)->create(); // Create fake 20 User in second  
App\Models\User::all(); App\Models\User::find(1); // Get Data from Database

GUv ‡Mj wUbKvi e¨envi K‡i WvUv ˆZwi wbqg GLb Avgiv wWwe wmWvi e¨envi K‡i wKfv‡e WvUv BbmvU© Kie †mUv we¯ÍvwiZ †`Le:

1. php artisan make:seeder UserSeeder   // Run This Command for make Seeder
2. **class** DatabaseSeeder **extends** Seeder
3. {
4. /\*\*
5. \* Run the database seeders.
6. \*
7. \* @return void
8. \*/
9. **public** **function** run()
10. {
11. DB::table('users')->insert([
12. 'name' => Str::random(10),
13. 'email' => Str::random(10).'@gmail.com',
14. 'password' => Hash::make('password'),
15. ]);
16. }

Gfv‡e bv nq wb‡Pi g‡Zv K‡i Gfv‡e wjLe |

1. **public** **function** run()
2. {
3. $faker = \Faker\Factory::create();
5. DB::table("users")->insert([
6. "name" => $faker->name(),
7. "email" => $faker->safeEmail,
8. "mobile" => $faker->phoneNumber,
9. "age" => $faker->numberBetween(25, 50),
10. "gender" => $faker->randomElement(["male", "female", "others"]),
11. "address\_info" => $faker->address,
12. ]);
13. }
14. }

Gfv‡e wj‡L wb‡ev me¸‡jv wdì Gi Z\_¨ | Gevi wb‡Pi g‡Zv K‡i mivmwi K¬vm a‡i wWwe wmW KgvÛ Pvjv‡Z cvwi | Avevi WvUv‡em wmWvi Gi wfZ‡i w`‡qI Pvjv‡Z cvwi wb‡Pi g‡Zv K‡i |

1. Running Seeder **class** directly
2. # Command to run seeder **class** file
4. $ php artisan db:seed --**class**=StudentSeeder

Using DatabaseSeeder to Seed data

Using DatabaseSeeder.php at /database/seeders/ folder

1. <?php
3. namespace Database\Seeders;
5. **use** Illuminate\Database\Seeder;
6. **use** Database\Seeders\StudentSeeder;
8. **class** DatabaseSeeder **extends** Seeder
9. {
10. /\*\*
11. \* Seed the application's database.
12. \*
13. \* @return void
14. \*/
15. **public** **function** run()
16. {
17. $this->call([
18. UserSeeder::**class**,
19. PostSeeder::**class**,
20. CommentSeeder::**class**, // For More Seeder Class
21. ]);
22. weKí Gfv‡eI †`Iqv hvq | wb‡Pi Uv n‡jv d¨v±ix e¨envi K‡i wmW Kiv Avi Dc‡i wmWvi dvBj ˆZwi K‡i |
23. \App\Models\Student::factory(100)->create();
24. User::factory()->count(50)->hasPosts(1)->create(); // GB KgvÛ BDRvi ‰Zwi mv‡\_ mv‡\_ cÖ‡Z¨K BDRv‡ii GKUv K‡i ‡cv÷ \_vK‡e Zvi Aax‡b GUvI Kiv hvq | GLv‡b †nR‡cv÷ gv‡b n‡jv †cv÷ g‡Wj Gi GKUv dvskb ˆZwi Kiv Av‡Q posts() bv‡g | ZvB GLv‡b Gfv‡e †`Iqv n‡q‡Q |
25. }
26. }
27. # Command to Seed Data
28. php artisan db:seed
29. The migrate:refresh --seed command is a shortcut to these 3 commands:
31. $ php artisan migrate:reset     # rollback all migrations
32. $ php artisan migrate           # run migrations
33. $ php artisan db:seed           # run seeders