

Laravel

১. layouts এর মধ্যে admin panel এবং website এর header এবং footer থাকে। যা @yield('content') দ্বারা page এর ভিতরের content কে ধারণ করে।

২. **php artisan make:auth** => এই command দিলে প্রথম page এ login and registration button চলে আসবে। (এটি larvae 6 এর নিচের জন্য প্রযোজ্য)

৩. **php artisan migrate** => .env file এ database connect করে এই command দিলে database এর মাঝে table create হয়। অনেক সময় এই command দিলে একটা error দিতে পারে। app-Providers->AppServiceProvider.php এ গিয়ে error টা solve করে নিচের command দিলে পুনরায় database এর সব table গুলো delete করে দিয়ে নতুন ভাবে সব table create করবে।

৪. **php artisan migrate:fresh** => এই command মূলত একটা database এর সকল table কে delete করে নতুন ভাবে সব table create করে। এ ক্ষেত্রে সব data delete হয়ে যাবে।

৫. **php artisan make:controller ControllerName** => এই command দ্বারা নতুন controller তৈরি করতে হয়।

৬. **php artisan make:model ModelName -m** => এই command দিলে নতুন model create হবে। সাথে একটি migration file create হবে।

৭. **php artisan make:controller ControllerName --model=ModelName** => Resource controller create করার জন্য এই command use করতে হয়। ModelName এর জায়গাতে যেই model এর নাম দিব সেই model টি আগেই create করে নিতে হয়। Resource controller create করলে ৭ টি function create হয়। এই ৭ টি function কে একটি route দ্বারা control করা যায়। route syntax:

```
Route::get('admin/user', 'UserController');
```

৮. **php artisan make:controller FolderName/ControllerName** => এই command দ্বারা একটি folder এর মাঝে controller create করা যায়। এক্ষেত্রে route syntax:

```
Route::get('admin/user', 'FolderName\UserController');
```

৯. Route group create করার নিয়মঃ

```
Route::group(['as' => 'admin.', 'prefix' => 'admin', 'namespace' => 'admin', 'middleware' => ['admin:admin', 'writer:writer']], function(){
```

```
Route::get('dashboard', 'DashboardController@index')->name('dashboard');
```

```
});
```

ব্যাখ্যাঃ

- 'as' => 'admin.' এর মানে হল name route এর মাঝে যা থাকবে সব গুলার আগে admin. বসবে। যেমনঃ admin.dashboard.
- 'prefix' => 'admin' এর মানে হল সকল url এর আগে admin বসবে। যেমনঃ admin/user, admin/create etc.
- 'namespace' => 'admin' এর মানে হল ওই route group এর মাঝে যে controller থাকবে সব গুলোই admin folder এর মাঝে আছে।

- 'middleware' => ['admin:admin', 'writer:writer'] middleware এর মাঝে যে middleware দিব সেটা শুধুমাত্র এই route group এর মাঝে অবস্থিত route গুলোকে protect করবে। এখানে admin:admin মানে হল প্রথম admin টা middleware আর দ্বিতীয় admin টা guard.

১০. **php artisan make:controller Admin/ControllerName -r** => এটিও resource controller create করার command. এই command দিলে শুধুমাত্র controller এর মাঝে তৈরিকৃত method গুলোর জন্য model এর object instance set করে দেয় না।

১১. **php artisan make:notification NotificationName** => প্রথম বার এই command দিলে project এর মাঝে notification নামে একটি folder create হবে এবং ওই folder এর মাঝে notification এর file create হবে।

১২. **composer remove packageName** => এই command এ packageName এর জায়গাতে যে package এর নাম দেওয়া হবে সেটি uninstall হয়ে যাবে।

১৩. .env file এর মাঝে APP_NAME এ যে নাম সেট করে দিব সেই নাম title এ show করবে। যদি name সেট করার পরে page reload দিয়ে title change না হয় তাহলে নিচের command দিতে হবে।

php artisan config:clear

Image Intervention

১. **composer require intervention/image** => Image upload এর জন্য এই package টা install করতে হবে। package ছাড়াও Image upload করা যায় তবে package দিয়ে Image upload করলে ভালো হয়।

২. config->app.php তে গিয়ে providers এর মাঝে দিতে হবে:

Intervention\Image\ImageServiceProvider::class

৩. আর aliases এর মাঝে দিতে হবে:

'Image' => Intervention\Image\Facades\Image::class

৪. **php artisan vendor:publish --**

provider="Intervention\Image\ImageServiceProviderLaravelRecent"

Note: কোন package install দিলে সেটা composer.json এর মাঝে "require:" এর মাঝে entry হয়ে থাকে। এটা আবার তুলে দিয়ে command line এ **composer update** command দিলেই composer গুলো update হয়ে যাবে। এবং vendor এর মাঝে যে file create হয় সেটাও delete করে দিলে ঐ package টা uninstall হয়ে যাবে।

৫. **php artisan storage:link** => এই command দিলে storage file এর একটি shortcut file public folder এর মাঝে create হয়, যার মাঝে image গুলো জমা থাকে।

Note: storage:link use না করে image upload করাই ভালো।

DB Seeder

১. `php artisan make:seed UsersTableSeeder` => এই command দিলে seed folder এর মাঝে UsersTableSeeder নামে একটি php file create হবে। এই file এর মাঝে default value গুলো set করে দেওয়া হয়।

Note: এই file এর মাঝের কাজ শেষ হলে অবশ্যই এই file কে DatabaseSeeder.php file এর মাঝে call করতে হবে।

২. `php artisan db:seed` => এই command দিলে file গুলো load হবে এবং database এ data input হবে। যদি কোন error দেয় তাহলে,

`composer dump-autoload` command run korte হবে। [important]

Laravel Project Create

Project Create:

১. প্রথমে composer দ্বারা একটি project create করতে হবে।

২. একটি নতুন database create করতে হবে।

৩. .env file এ database name টা add করে দিতে হবে।

Basic Authentication:

১. `composer require laravel/ui:^2.4` => এন্স একটি command দিতে হবে।

২. `php artisan ui vue --auth` => এই command দিতে হবে।

৩. `npm install` => এই command দিতে হবে। যার ফলে project এ node_modules নামে একটি file create হবে।

Note: pc তে যদি node js install না দেওয়া থাকে তাহলে উপরের command টি কাজ করবে না সেই জন্য pc তে অবশ্যই node js install দেওয়া থাকতে হবে। node js install দেওয়া হলে pc তে node js এর path set করে দিতে হয়। path set করা হলে pc অবশ্যই restart দিতে হবে।

৪. `npm run dev` => এই command দিলেই finally শেষ।

Laravel Relationship

১. One To Many (hasMany):

Syntax: `return $this->hasMany('App\ModelName', 'foreign_key');`

Note: যদি foreign_key এর নাম model এর নামে দেই তাহলে parameter এর মাঝে foreign_key বলে দিতে হয় না। নিচের চিত্রের মত।

২. One To Many Inverse (belongsTo):

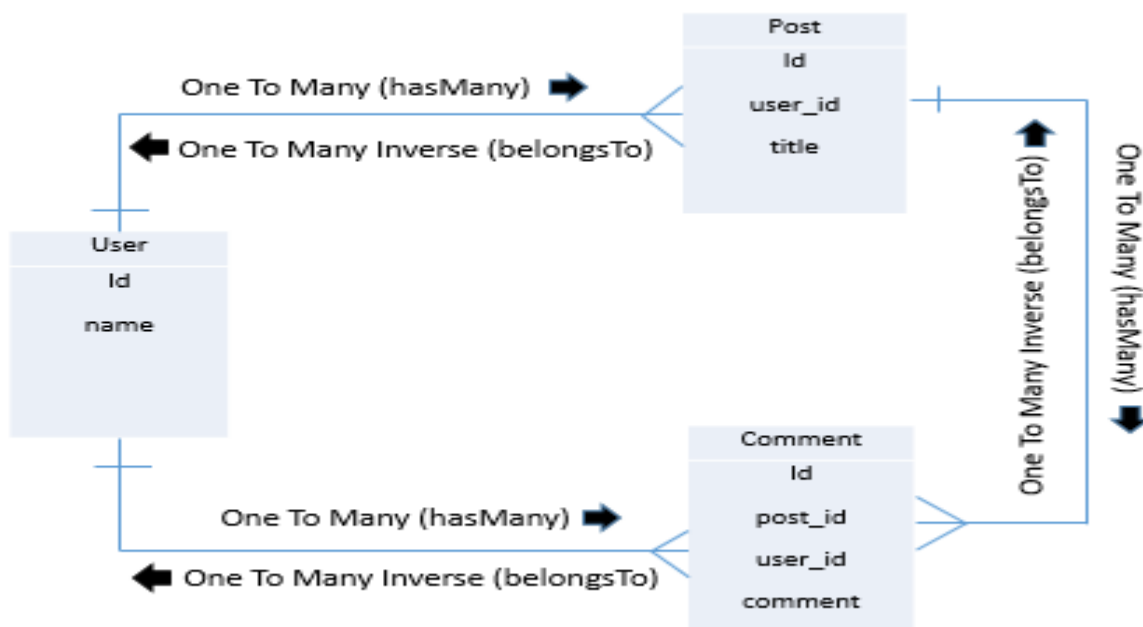
Syntax: `return $this->belongsTo('App\ModelName', 'foreign_key');`

Note: যদি `foreign_key` এর নাম `model` এর নামে দেই তাহলে `parameter` এর মাঝে `foreign_key` বলে দিতে হয় না। নিচের চিত্রের মত।

Note: One To Many Inverse হুবহু One To One Relation এর মত কাজ করে।

উদাহরণঃ যখন একটি `table` এ অন্য একটি `table` এর `ID` `foreign_key` হিসাবে থাকে তখন ঐ `table` দ্বয়ের মাঝে One To Many এবং One To Many (Inverse) relation কাজ করে।

Note: যেখানে One To Many relation থাকবে সেখানে অবশ্যই One To Many (Inverse) relation ও থাকবে।



৩. Many To Many (belongsToMany):

Syntax: `return $this->belongsToMany('App\ModelName')->withTimestamps();`

Note: `withTimestamps()` function টা না দিলে pivot table এর `created_at` এবং `updated_at` column টা ফাঁকা থাকে।

যখন একটি `table` এর একাধিক records অন্য একটি `table` এর একাধিক records এর সম্পর্কিত তখন Many To Many relation এর প্রয়োজন পড়ে।

উদাহরণঃ post এবং category দুইটি table এর মাঝে Many To Many relation হতে পারে। যেমন, একটি post এর অনেক গুলো category থাকতে পারে আবার একটি category এর অনেক গুলো post থাকতে পারে।

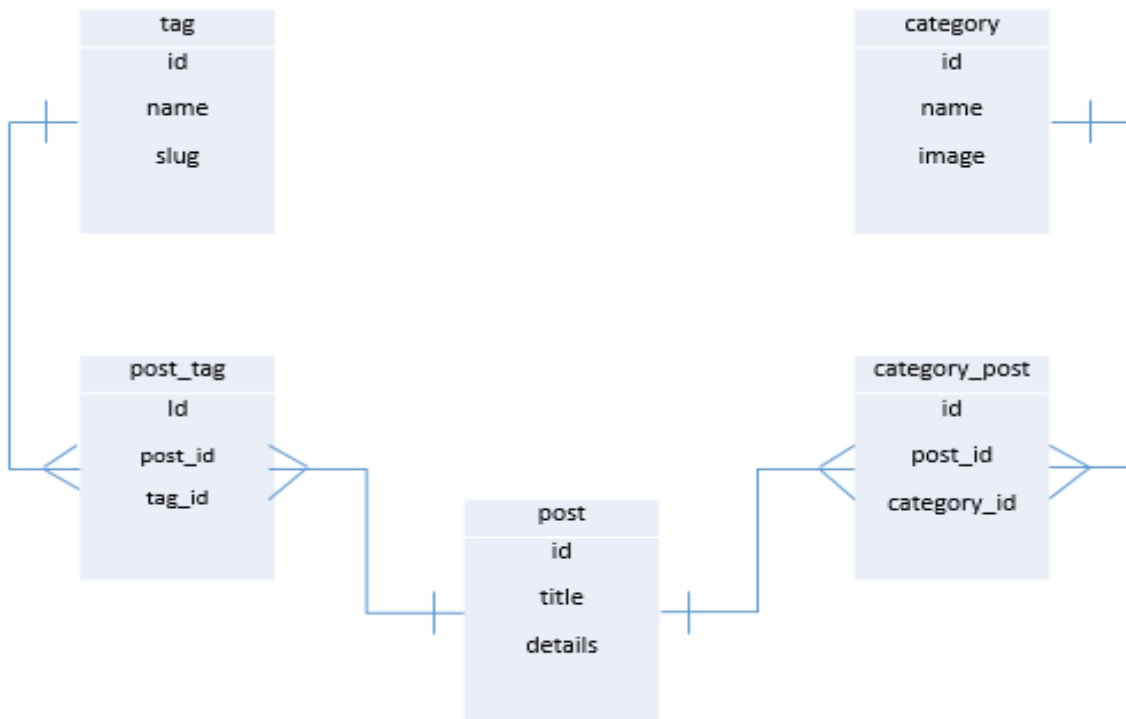
Many To Many relation এর শর্তঃ

১. প্রতিটি Many To Many relation এর ক্ষেত্রে একটি pivot table এর প্রয়োজন পড়ে। যে দুইটি table এর মাঝে Many To Many relation হয় সেই দুইটি table এর ID গুলোকে শুধুমাত্র pivot table এ store করা হয়।

২. Pivot table এর কোন model create করতে হয় না শুধুমাত্র migration file create করতে হয়।

৩. Pivot table এর নাম alphabetic order এ হয়। যদি post এবং category table এর pivot table তৈরি করতে হয় তাহলে নাম হবে category_post.

Example: php artisan make:migration create_category_post_table



MVC:

M = Model (Database)

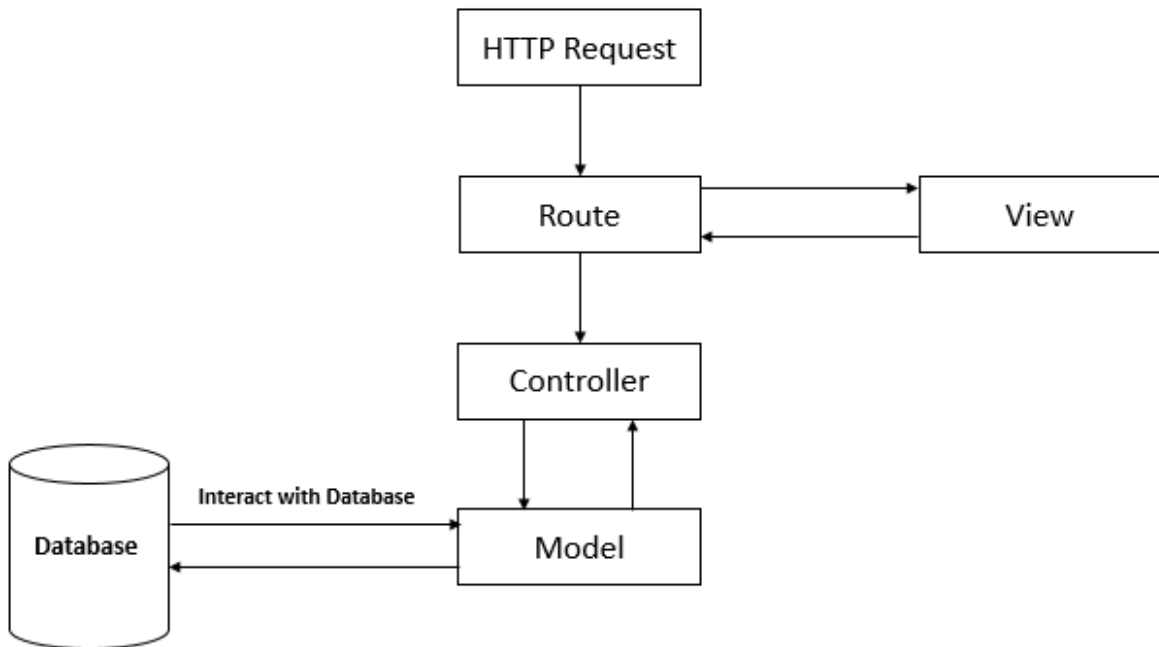
V = View (আমরা যা দেখি)

C = Controller (যেখানে আমরা logic লিখি)

HTTP Request:

URL এ যে request যায় তাকেই http request বলে। যেমনঃ যখন আমরা login button এ click করি তখন আমাদের login page এ নিয়ে যায় এবং URL এ একটি change আসে। এটাই http request. অনেক ধরনের http request আছে। যথাঃ

1. GET
2. POST
3. PUT
4. PATCH
5. DELETE



যখন কোন http request করা হয় তখন request টা প্রথমে যায় route এ তারপরে যায় controller এ, Then controller check করে, যে request টি দেওয়া হয়েছে সেটাতে database থেকে কোন data নিয়ে আসতে হবে বা database এ data পাঠাইতে হবে কি না। যদি database এ কাজ থাকে তাহলে

সেই কাজ করার জন্য model এ যায়। তারপর model থেকে আবার controller এ যায়। তারপর controller থেকে view page এ যায়। যদি database এর কোন কাজ না থাকে তাহলে controller request টা model এ পাঠায় না সরাসরি view page show করায়।

MVT For Django:

M = Model

V = View

T = Template

Note: Django মূলত model দিয়েই সব কাজ করে।