Laravel Notes

**HOW TO(S)**

1. **How to create a model and its database table  
   (e.g. Profile model)?**
2. *Open CMD from project folder.*
3. *Type –* ***Php artisan***
4. *Write the following:****php artisan make:model Profile –m*** *“-m” will create a table named “profiles” and a php file “create\_profiles\_table.php”*
5. *Go to “create\_profiles\_table.php” and create all the necessary rows with appropriate data types and properties.*
6. *Go to Php artisan in CMD terminal and type the following:****Php artisan migrate***

***Note:***

* *Refer to* ***1:10:21*** *of the video****.***

**2. How to create a controller.php file for a model?**

**Relationship Implementation:**

**1. How to implement M-to-M pivot relationship between two models (e.g User-Profile Following)?**

*a. Go to resources->js->app.js and create a vue.component*

*file(e.g. FollowButton.vue).*

*b. Inside FollowButton.vue template, add the <button>.*

*c. create a vue button e.g <Follow-Button></Follow-Button> in*

*the blade file.*

*d. Run npm run watch on CMD.*

*e. Add the appropriate method, data, computed-button in FollowButton.vue.*

*f. Create a controller such as FollowController.php to operate functions forwared from FollowButton.vue's method.*

*g. create a restful route in web.php that forwards the method to*

*the controller functions.*

*h. create a pivotal table in php artisan*

*(e.g. php artisan make:migrate creates\_profile\_user\_pivot\_table --create profile\_user)*

*make sure to follow naming convention in alphabetical order when defining models here.*

*i. Make the schema (e.g.)*

*public function up()*

*{*

*Schema::create('profile\_user', function (Blueprint $table) {*

*$table->bigIncrements('id');*

*$table->unsignedBigInteger('profile\_id');*

*$table->unsignedBigInteger('user\_id');*

*$table->timestamps();*

*});*

*}*

*j. Migrate your database (e.g php aritsan migrate).*

*k. create a M-to-M relationship specifier function in User and Profile class e.g.*

*public function following()*

*{*

*return $this->belongsToMany(Profile::class);*

*}*

*l. Now toggle the relationship using the FollowController.php.*

***Note:***

* *When implementing M-to-M relationship among models a pivot table is required to hold* ***ID*** *for both the model entity.*
* *Flow of operation:   
  Authenticated user click Follow button 🡪 followUser() operates; calls the store() in FollowController.php using restful Api then toggle status(true/false) 🡪 store() toggles between following() in User class and followers() in profile class thus storing the data in profile\_user table(pivot) 🡪 buttonText() returns value based on altered status.*

**2. How to implement O-to-M relationship between two models (e.g One user has many posts)?**

1. *Create a public function in User.php that returns all the post objects a user has.*

*e.g-  
 public function posts(){*

*return $this->hasMany(Post::class)->orderBy('created\_at', 'DESC');*

*}*

1. *Create a public function in Post.php that returns the user object where a specific post belongs to.*

*e.g-  
 public function user()*

*{*

*return $this->belongsTo(User::class);*

*}*

**3. How to implement O-to-O relationship between two models (e.g One user has one profile)?**

1. *Create a public function in User.php that returns the profile object a user has.*

*e.g-*

*public function profile(){*

*return $this->hasOne(Profile::class);*

*}*

1. *Create a public function in Profile.php that returns the user object a profile belongs to.*

*e.g-*

*public function user(){*

*return $this->belongsTo(User::class); // creating O-to-O relationship with User class*

*}*

**Basic Php Artisan commands**

1. ***Php artisan serve :*** *shows all the artisan commands with explanation.*

1. ***Php artisan tinker :*** *Shows relationships between multiple models*

*( e.g. Posts that a specific user has  
$u=User::find(1);  
$u->Posts;*

*)*

1. ***Implementing Policy on Models :***  *Code example: Php artisan make:policy ProfilePolicy –m Profile**This is going to make a php file named ProfilePolicy.php that applies policies on model Profile. Policies include if a user can restore, update, delete a Profile entity based on the functions that return Boolean values.*
2. *Whenever update, delete or edit make sure to code something like:* ***$this->authorize('update', $user->profile )  
   auth()->user()->profile->update()*** *Otherwise unauthorized user can access and alter personal data through incognito web mode.*

**Codes and Function Explanation**

1. ***public function \_\_construct():*** *Each Controller file starts with this function. Whatever functions that come after this function has to ensure that the current user is authorized, otherwise these functions wont operate rather redirect to the login page.*
2. ***Create():*** This function accepts an array of attributes, creates a model entity, and inserts it into the database.

*For example :*

auth()->User()->posts()->create([

'caption' => $data['caption'],

'image' => $image\_path,]);

1. **findorfail():**findOrFail($id) takes an id and returns a single model. If no matching model exist, it throws an error.
2. **WhereIn() :** Laravel provide wherein() to use sql wherein query. in wherein() we just need to pass two argument one is column name and another if array of ids or anything

**TO DO(s):**

1. Solution for printing each comment’s reacting users:  
   Whenever the Likes link is clicked query the database for that comment’s reactors and print them on the modal.
2. When printing total number of likes on the Love-React button set an **if** condition . If number of reactor is 1 print “1 Like” else “3 *or*  4 Likes”.