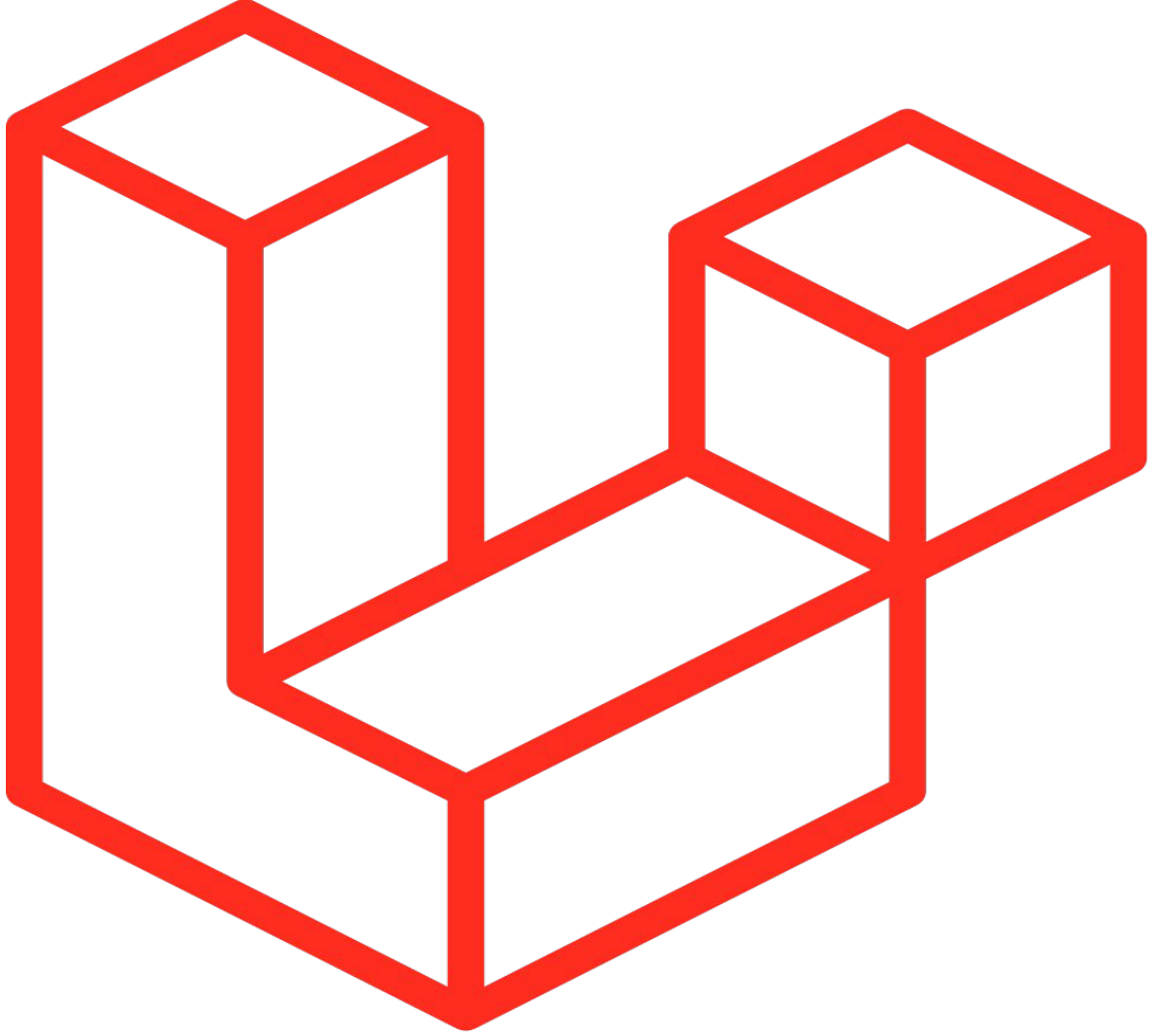
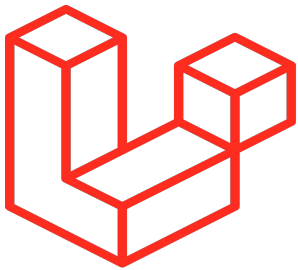


Content

- DB Facade & Eloquent
- Database Factory
- Database Seeds
- Artisan Tinker



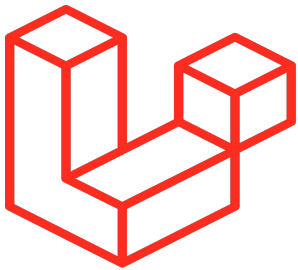
DB Facade & Eloquent

- $post = DB::table(‘posts’)->find(20); //finds a row with id = 20

- $posts = DB::table('posts')->get(); //get all rows in posts table

- $singlePost = DB::table(‘posts’)->where(‘slug’ , ‘FirstPost’)->first(); //where conditions to query

- $firstPost = DB::table(‘posts’)->first(); //gets the first row

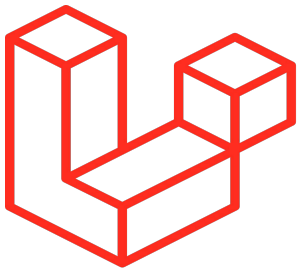


DB Facade & Eloquent

- DB::table(‘posts’)->insert(
  [‘title’ => ‘first post title’ , ‘desc’ => ‘first post desc ‘]
  ); //inserting a row

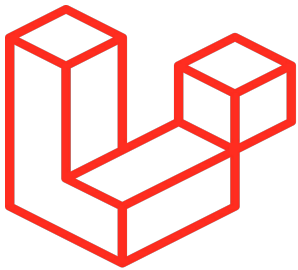
- DB::table('posts')->where(‘id’ , 1)->delete(); //deletes a row

- DB::table(‘posts’)->where(‘id’ , 1)
  ->update([‘title’ => ‘changed title post]);
  //update the post title only



DB Facade & Eloquent

- Laravel ORM is called Eloquent
- Database configuration in .env file or from config/database.php.
- Laravel migration helps in making database persistent across multiple machines .
- DB facade used to form query builder object



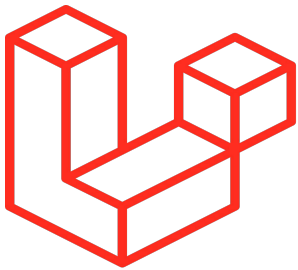
DB Facade & Eloquent

- php artisan make:model Post //create a new model class

- Laravel by default gets the plural name of model as a table name and makes query based on that

- Post::all() //will search in posts table and get all rows

- Post::create([‘title’ => ‘first post’ , ‘desc’ => ‘desc post’);
  //this will give a MassAssignmentException unless you override $fillable



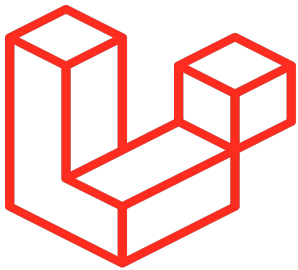
DB Facade & Eloquent

- Post::find(25)->update([‘title’ => ‘update post title’)

  //updates title for post with id 25

- Post::where(‘votes’, 23)->delete()

  //deletes any post have votes with 23



DB Facade & Eloquent

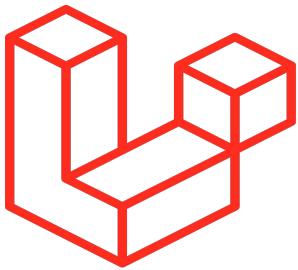
- To define a relation in Post model you will define a function in the class for example i want to say post have many sections .

```
//in Post model class

public function sections ()

{ return $this->hasMany(Section::class);}

// in controller for example
$postSections = Post::find(23)->sections;
```

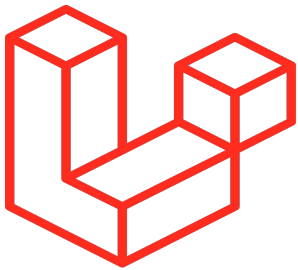


DB Facade & Eloquent

- Remember that query results are collection objects

- More at Eloquent
  https://laravel.com/docs/master/eloquent#introduction

- More at Collections
  https://laravel.com/docs/master/collections#available-methods

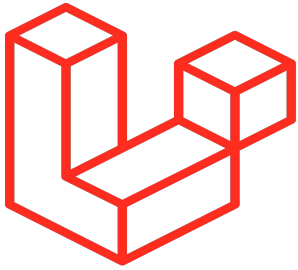


Database Factory

- Laravel model factories and seeders make it painless to create test database records using your application's Eloquent models and relationships

- php artisan make:factory PostFactory

  https://laravel.com/docs/master/database-testing#concept-overview

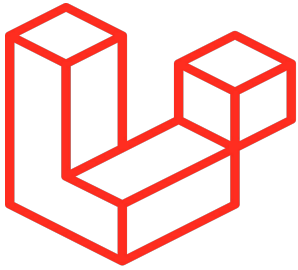


Database Seeds

- **Use model factories to conveniently generate large amounts of database records.**

- **php artisan make:seeder PostSeeder**

- **Inside PostSeeder we will use PostFactory**
  https://laravel.com/docs/master/seeding#using-model-factories

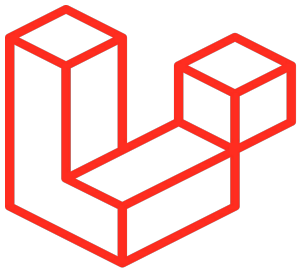


Artisan Tinker

- **Tinker allows you to interact with your entire Laravel application on the command line, including your Eloquent models, jobs, events, and more. To enter the Tinker environment, run the `tinker` Artisan command:**

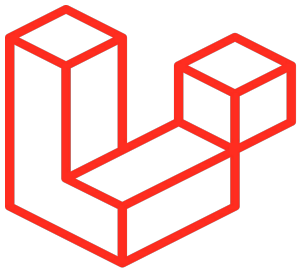
- **php artisan tinker**

  https://laravel.com/docs/master/artisan#usage



Lab 2

- Create migrations & model for the necessary db posts table
- make sure CRUD operation on Posts are stored in the DB
- When i Click on Delete you must show a warning before deleting and i choose between yes to confirm Delete or no … and you must use Route::delete
- In Index & Show page ,make sure the Created At is formated , so read carbon documentation https://carbon.nesbot.com/docs/
- In Edit Or Create Post Creator Field must be drop down list of users
- Create PostSeeder & PostFactory class so that when i run php artisan db:seed it seeds posts table with 500 records
- Add pagination to Index page Read About Pagination then display pagination links
- Add CURD comments inside show post page using polymorphic relation https://laravel.com/docs/master/eloquent-relationships#polymorphic-relationships … don't overengineer this one and use ajax requests … just simple form submissions



Lab 2 (Bonus)

- Add restore button on index page to restore deleted posts you will need to use soft delete https://laravel.com/docs/master/eloquent#soft-deleting

- create Accessor Method inside Post Model that returns human readable carbon to be used in posts/{id} , for example i want $post->human_readable_date will result in the formated carbon date that is rendered in show post page https://laravel.com/docs/master/eloquent-mutators#defining-an-accessor

- Add View Ajax Button to posts page , that opens Bootstrap Modal , showing post info (title , description , username, useremail) using ajax request (check laravel responsable interface in docs to make your code cleaner)

- Use livewire to make your comments doesn't refresh the page when making CRUD

  first check this video to understand what is livewire

  then check this livewire comments component video in case it may be helpful

Disclaimer: livewire is so powerful to give you the feel of SPA without the complexity of frontend frameworks . but it has some limitations we can discuss this more in next lecture