1. Untuk membuat migration dengan nama posts, Anda dapat menggunakan perintah artisan berikut:

php artisan make:migration create\_posts\_table

2. Setelah migration dibuat, buka file migration yang telah dihasilkan di dalam direktori database/migrations. Tambahkan kolom-kolom sesuai dengan entitas yang Anda butuhkan seperti berikut:

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreatePostsTable extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('posts', function (Blueprint $table) {

$table->id();

$table->string('title');

$table->string('description');

$table->string('category');

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('posts');

}

}

Dengan script di atas, Anda akan membuat tabel posts yang memiliki kolom title, description, dan category, semuanya dengan tipe data string. Jangan lupa untuk menjalankan perintah php artisan migrate untuk menerapkan migration tersebut.

3. Mengirimkan migration untuk membuat tabel posts ke dalam database:

php artisan migrate

4. Membuat model dengan nama Post:

php artisan make:model Post

5. Deklarasikan nama tabel posts dalam model Post. Buka file Post.php di direktori app/Models dan tambahkan deklarasi berikut:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Post extends Model

{

use HasFactory;

protected $table = 'posts';

// Deklarasikan nama-nama kolom yang dapat diisi (mass assignable)

protected $fillable = ['title', 'description', 'category'];

}

7. php artisan make:migration create\_comments\_table

8. Membuat entitas migration pada comments. Buka file migration yang telah dihasilkan di dalam direktori database/migrations dan tambahkan kolom-kolom sesuai dengan entitas yang Anda butuhkan seperti berikut:

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateCommentsTable extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('comments', function (Blueprint $table) {

$table->id();

$table->string('content');

$table->foreignId('user\_id')->constrained()->onDelete('cascade');

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('comments');

}

}

9. Mengirimkan migration untuk membuat tabel comments ke dalam database:

php artisan migrate

10. Membuat model dengan nama Comment:

php artisan make:model Comment

11. Deklarasikan nama tabel comments dalam model Comment. Buka file Comment.php di direktori app/Models dan tambahkan deklarasi berikut:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Comment extends Model

{

use HasFactory;

protected $table = 'comments';

// Deklarasikan nama-nama kolom yang dapat diisi (mass assignable)

protected $fillable = ['content', 'user\_id'];

}

12. Deklarasikan field atau attribute nama-nama kolom pada tabel comments yang telah dibuat dalam model Comment:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Comment extends Model

{

use HasFactory;

protected $table = 'comments';

// Deklarasikan nama-nama kolom yang dapat diisi (mass assignable)

protected $fillable = ['content', 'user\_id'];

}

13. Membuat PostController:

php artisan make:controller PostController

14. Membuat CommentController:

php artisan make:controller PostController

15. Buat 1 fungsi index dalam PostController. Buka file PostController.php di direktori app/Http/Controllers dan tambahkan fungsi berikut:

namespace App\Http\Controllers;

use App\Models\Post;

use Illuminate\Http\Request;

class PostController extends Controller

{

public function index()

{

$posts = Post::with('comments')->get();

return view('posts.index', compact('posts'));

}

}

16. Buat 1 fungsi index dalam CommentController. Buka file CommentController.php di direktori app/Http/Controllers dan tambahkan fungsi berikut:

namespace App\Http\Controllers;

use App\Models\Comment;

use Illuminate\Http\Request;

class CommentController extends Controller

{

public function index()

{

$comments = Comment::with('post')->get();

return view('comments.index', compact('comments'));

}

}

17. Buat 1 route untuk mengambil URL dengan nama admin-galeri-photo beserta CommentController, nama function index. Buka file web.php di direktori routes dan tambahkan route berikut:

use App\Http\Controllers\CommentController;

Route::get('admin-galeri-photo', [CommentController::class, 'index']);

19. Buat 1 route untuk mengambil URL dengan nama user-comment-photo beserta PostController, nama function index. Buka file web.php di direktori routes dan tambahkan route berikut:

use App\Http\Controllers\PostController;

Route::get('user-comment-photo', [PostController::class, 'index']);

19. Buat function dengan nama comments dalam class model Post untuk membuat function relasi antara class Post ke class Comment yang memiliki relasi one to many dari tabel posts ke tabel comments. Tambahkan fungsi berikut dalam model Post:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Post extends Model

{

use HasFactory;

protected $table = 'posts';

// Deklarasikan nama-nama kolom yang dapat diisi (mass assignable)

protected $fillable = ['title', 'description', 'category'];

public function comments()

{

return $this->hasMany(Comment::class);

}

}

20. Buat function dengan nama post dalam class model Comment untuk membuat function relasi antara class Comment ke class Post yang memiliki relasi one to many dari tabel comments ke tabel posts. Tambahkan fungsi berikut dalam model Comment:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Comment extends Model

{

use HasFactory;

protected $table = 'comments';

// Deklarasikan nama-nama kolom yang dapat diisi (mass assignable)

protected $fillable = ['content', 'user\_id'];

public function post()

{

return $this->belongsTo(Post::class);

}

}

21. Inisialisasi class objek dari class Post dalam function index di PostController serta cantumkan nama function comments menggunakan nama function ::with() dan dilanjutkan dengan function get():

namespace App\Http\Controllers;

use App\Models\Post;

use Illuminate\Http\Request;

class PostController extends Controller

{

public function index()

{

$posts = Post::with('comments')->get();

return view('posts.index', compact('posts'));

}

}

22. Inisialisasi class objek dari class Comment dalam function index di CommentController serta cantumkan nama function post menggunakan nama function ::with() dan dilanjutkan dengan function get():

namespace App\Http\Controllers;

use App\Models\Comment;

use Illuminate\Http\Request;

class CommentController extends Controller

{

public function index()

{

$comments = Comment::with('post')->get();

return view('comments.index', compact('comments'));

}

}