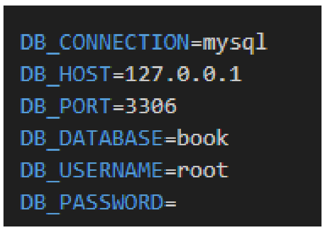
LABORATORY ACTIVITY: Populating From a Database

Dollesin, Karylle M.
BSIT 3C

DATABASE:

We created a table named ‘book’ in our chosen MySql server, Xampp.

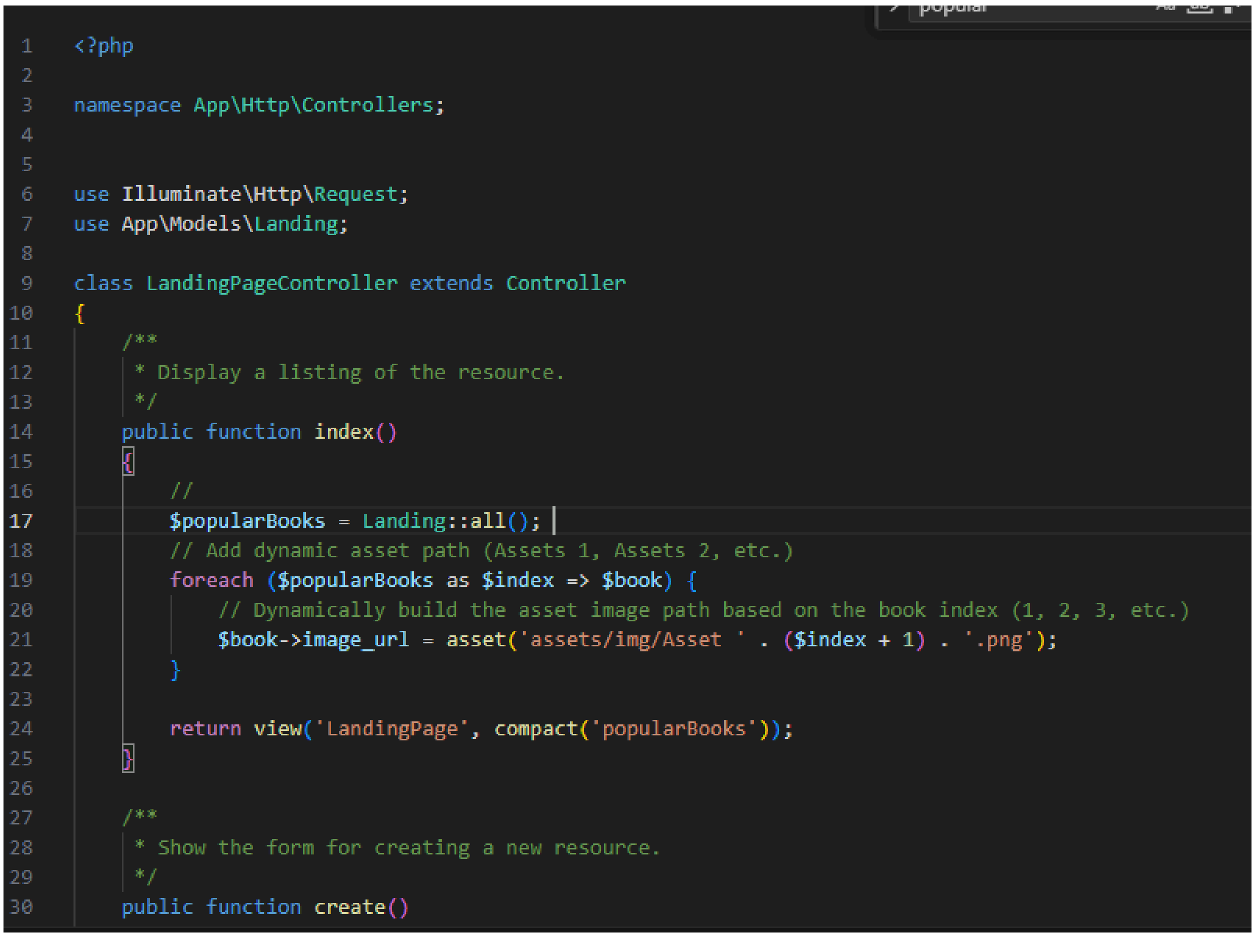
.env:



```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=book
DB_USERNAME=root
DB_PASSWORD=
```

We modified the .env file to update the necessary configuration settings for our application.

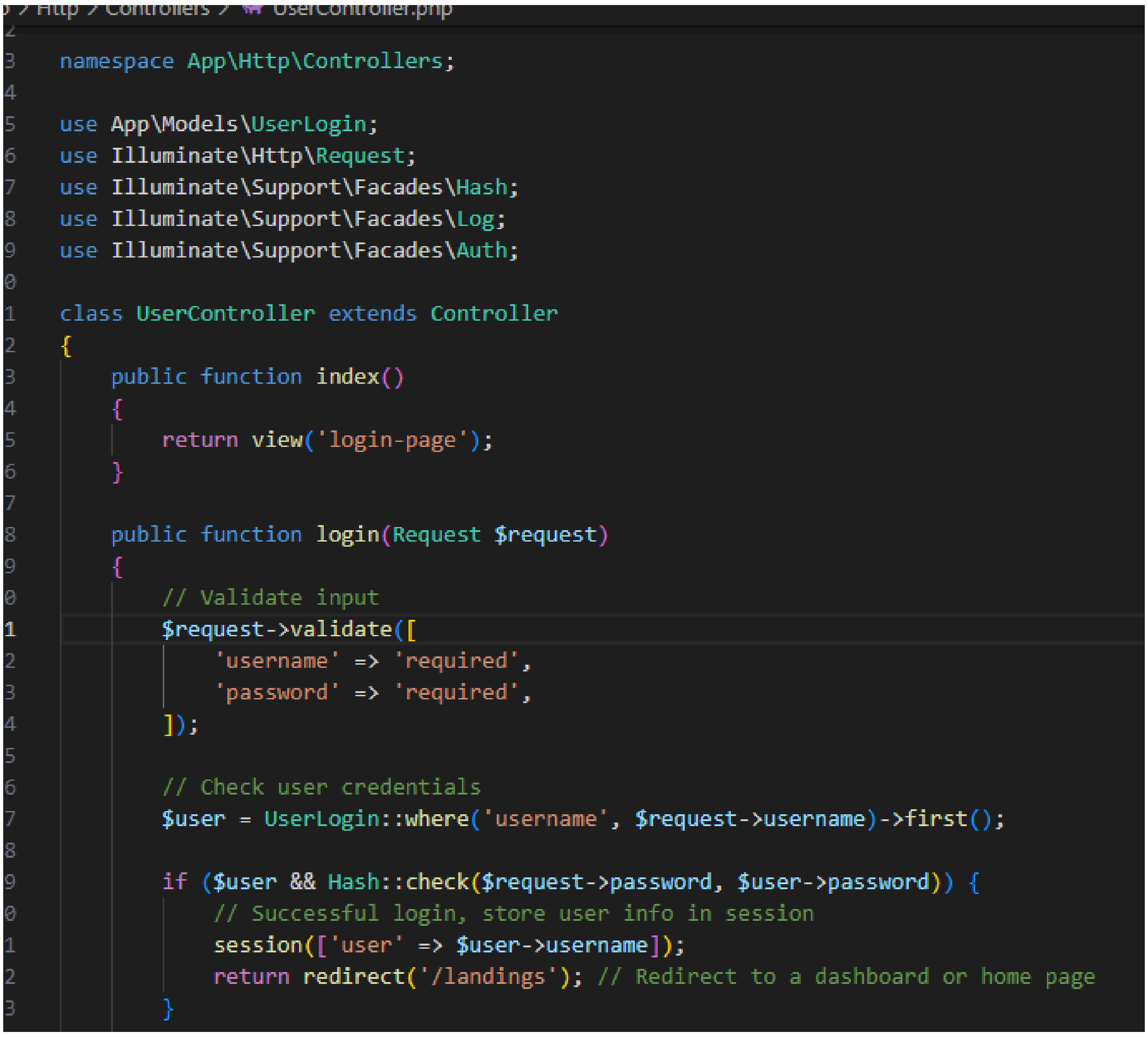
a. CONTROLLERS: LandingPageController.php



```php
<?php

namespace App\Http\Controllers;


use Illuminate\Http\Request;
use App\Models\Landing;

class LandingPageController extends Controller
{
    /**
     * Display a listing of the resource.
     */
    public function index()
    {
        //
        $popularBooks = Landing::all();
        // Add dynamic asset path (Assets 1, Assets 2, etc.)
        foreach ($popularBooks as $index => $book) {
            // Dynamically build the asset image path based on the book index (1, 2, 3, etc.)
            $book->image_url = asset('assets/img/Asset ' . ($index + 1) . '.png');
        }

        return view('LandingPage', compact('popularBooks'));
    }

    /**
     * Show the form for creating a new resource.
     */
    public function create()
```

We created a resource controller ‘LandingPageController’, this controller manages the CRUD operations related to book resources. The index() uses a function all() to retrieve all records from the database and store them in ‘popularBooks’.

b. CONTROLLERS: UserController.php



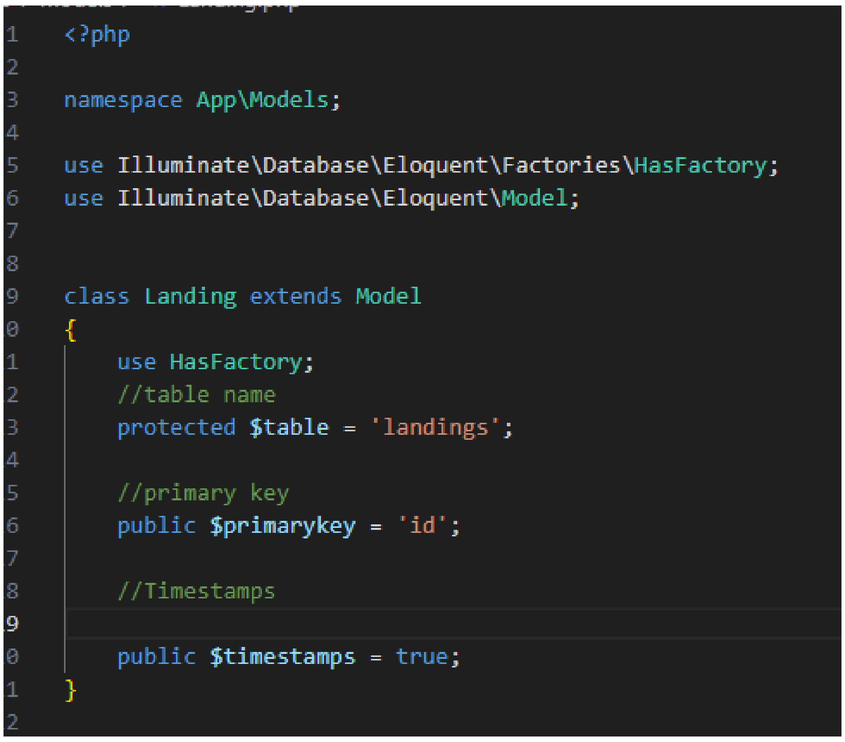
```php
namespace App\Http\Controllers;

use App\Models\UserLogin;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Log;
use Illuminate\Support\Facades\Auth;

class UserController extends Controller
{
    public function index()
    {
        return view('login-page');
    }

    public function login(Request $request)
    {
        // Validate input
        $request->validate([
            'username' => 'required',
            'password' => 'required',
        ]);

        // Check user credentials
        $user = UserLogin::where('username', $request->username)->first();

        if ($user && Hash::check($request->password, $user->password)) {
            // Successful login, store user info in session
            session(['user' => $user->username]);
            return redirect('/landings'); // Redirect to a dashboard or home page
        }
```

This controller handles User Login and Authentication. The index method renders the Blade view for the login-page route, displaying the login form.

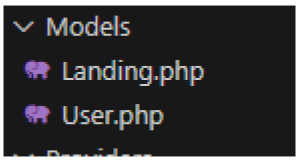
When the user submits the login form, the login method retrieves the username and password entered by the user. It interacts with the UserLogin model, which connects to the user_logins table in the database, to verify the credentials.

If the entered credentials match a record in the database, the user is redirected to the website's homepage.

c. MODELS: Landing.php



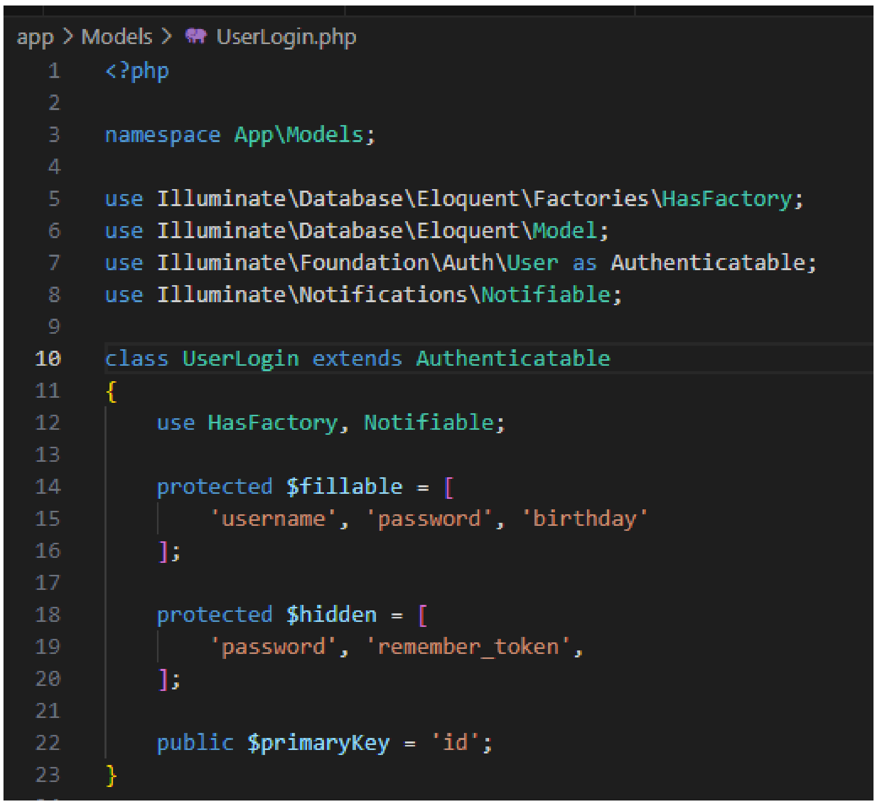
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;


class Landing extends Model
{
    use HasFactory;
    //table name
    protected $table = 'landings';

    //primary key
    public $primarykey = 'id';

    //Timestamps

    public $timestamps = true;
}

```



Models
Landing.php
User.php

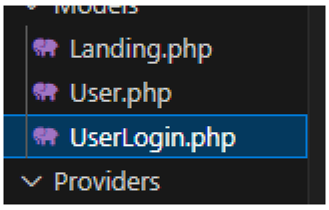
The Landing model maps to the 'landings table', uses id as its primary key, and manages timestamps (created_at and updated_at) automatically.

d. MODELS: UserLogin.php



app > Models > UserLogin.php

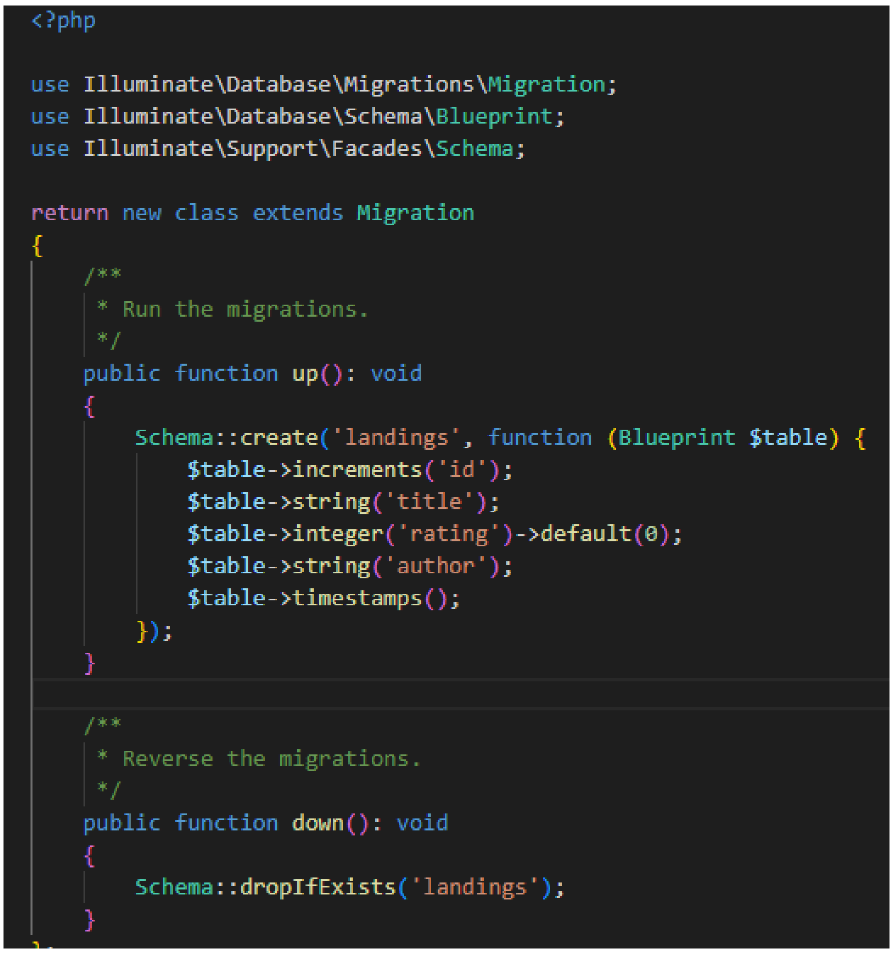
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;

class UserLogin extends Authenticatable
{
    use HasFactory, Notifiable;

    protected $fillable = [
        'username', 'password', 'birthday'
    ];

    protected $hidden = [
        'password', 'remember_token',
    ];

    public $primaryKey = 'id';
}
```



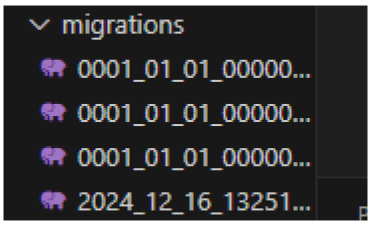
Landing.php
User.php
UserLogin.php
Providers

This model handles user data. It uses a built-in authentication features, and uses the HasFactory trait for database seeding and Notifiable traits for notification. The $fillable property ensures secure mass assignment for username, password, and birthday fields, while the $hidden property hides password and remember_token for security purposes.

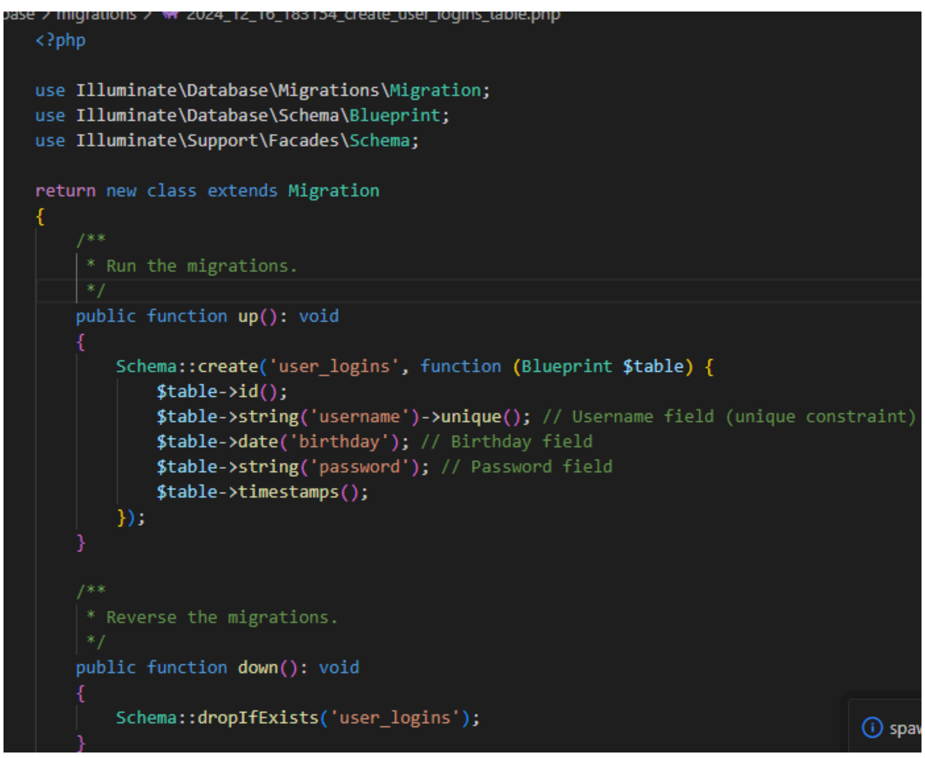
e. MIGRATIONS: landings & user_logins



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('landings', function (Blueprint $table) {
            $table->increments('id');
            $table->string('title');
            $table->integer('rating')->default(0);
            $table->string('author');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('landings');
    }
```

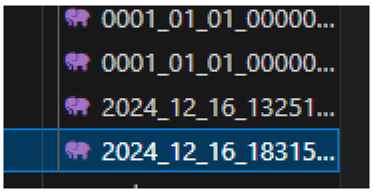


```
migrations
  0001_01_01_00000...
  0001_01_01_00000...
  0001_01_01_00000...
  2024_12_16_13251...
```



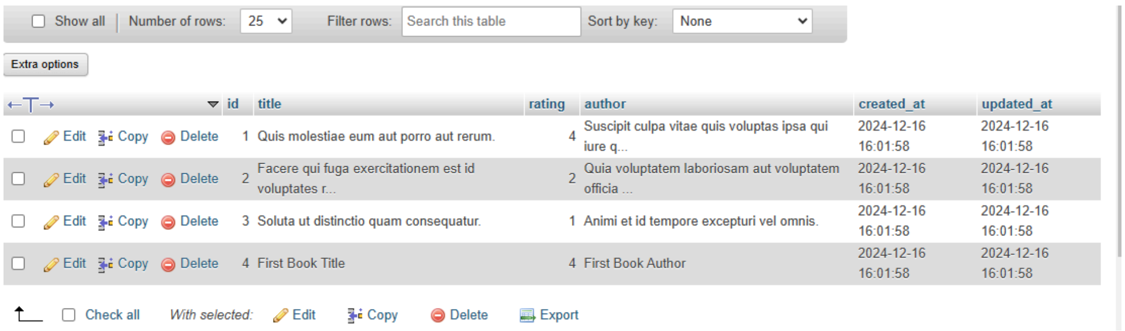
base > migrations > 2024_12_16_183154_create_user_logins_table.php

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('user_logins', function (Blueprint $table) {
            $table->id();
            $table->string('username')->unique(); // Username field (unique constraint)
            $table->date('birthday'); // Birthday field
            $table->string('password'); // Password field
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('user_logins');
    }
```



```
0001_01_01_00000...
0001_01_01_00000...
2024_12_16_13251...
2024_12_16_18315...
```

When ‘php artisan migrate’ runs it will create tables in the database. In this case ‘landings’ and ‘user_logins’ are the name of the tables with their respective columns.

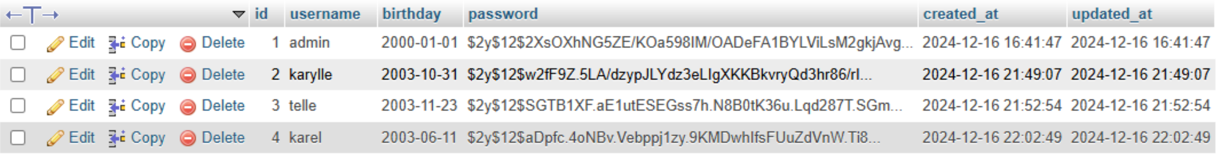


□ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

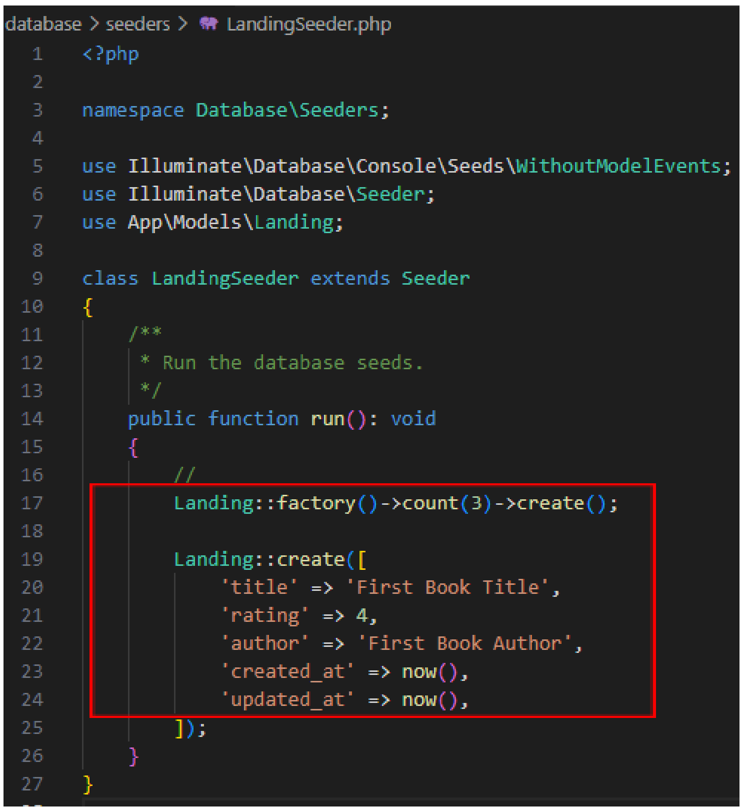
←T→	id	title	rating	author	created_at	updated_at
□ Edit Copy Delete	1	Quis molestiae eum aut porro aut rerum.	4	Suscipit culpa vitae quis voluptas ipsa qui iure q...	2024-12-16 16:01:58	2024-12-16 16:01:58
□ Edit Copy Delete	2	Facere qui fuga exercitationem est id voluptates r...	2	Quia voluptatem laboriosam aut voluptatem officia ...	2024-12-16 16:01:58	2024-12-16 16:01:58
□ Edit Copy Delete	3	Soluta ut distinctio quam consequatur.	1	Animi et id tempore excepturi vel omnis.	2024-12-16 16:01:58	2024-12-16 16:01:58
□ Edit Copy Delete	4	First Book Title	4	First Book Author	2024-12-16 16:01:58	2024-12-16 16:01:58

□ Check all *With selected:* Edit Copy Delete Export



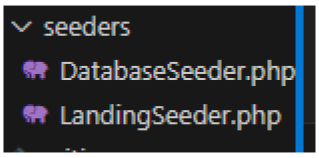
←T→				id	username	birthday	password	created_at	updated_at
□	Edit	Copy	Delete	1	admin	2000-01-01	$2y$12$2XsOXhNG5ZE/KOa598IM/OADeFA1BYLViLsM2gkjAvg...	2024-12-16 16:41:47	2024-12-16 16:41:47
□	Edit	Copy	Delete	2	karylle	2003-10-31	$2y$12$w2fF9Z.5LA/dzypJLYdz3eLIgXKKBkvryQd3hr86/rl...	2024-12-16 21:49:07	2024-12-16 21:49:07
□	Edit	Copy	Delete	3	telle	2003-11-23	$2y$12$SGTB1XF.aE1utESEGss7h.N8B0tK36u.Lqd287T.SGm...	2024-12-16 21:52:54	2024-12-16 21:52:54
□	Edit	Copy	Delete	4	karel	2003-06-11	$2y$12$aDpfc.4oNBv.Vebppj1zy.9KMDwhlfsFUuZdVnW.Ti8...	2024-12-16 22:02:49	2024-12-16 22:02:49

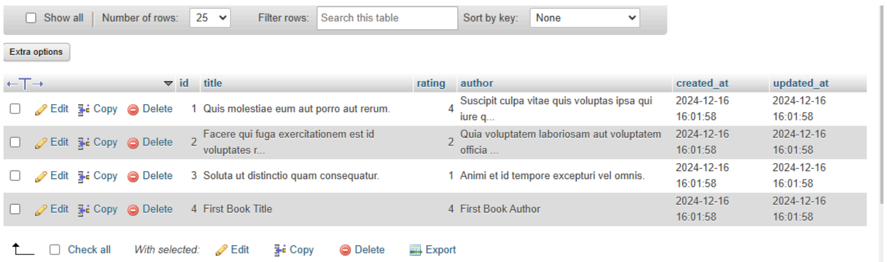
f. SEEDING: LandingSeeder.php and UserLoginSeeder.php



database > seeders > LandingSeeder.php

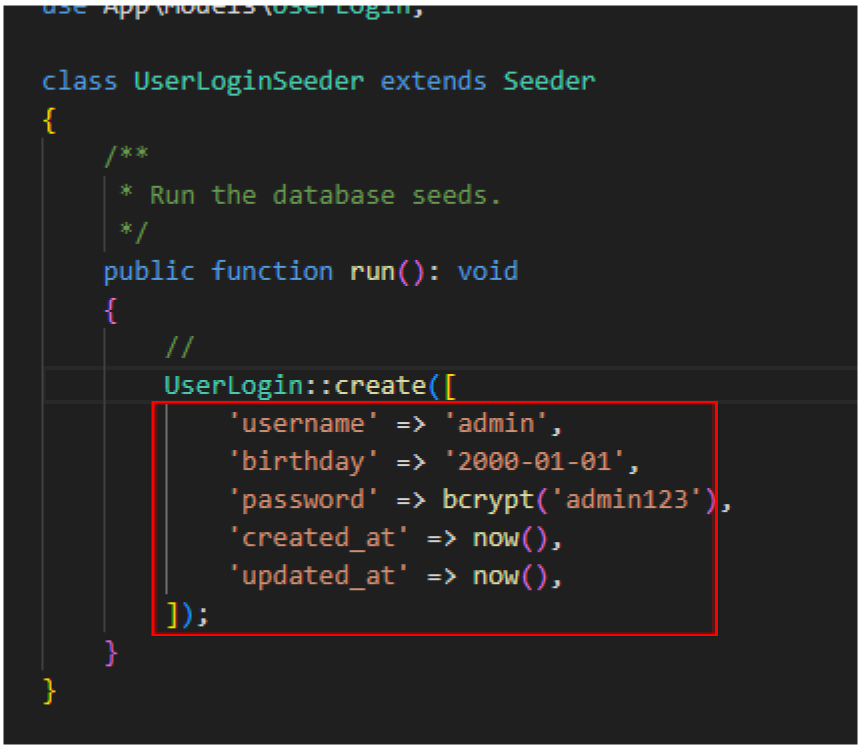
```php
<?php

namespace Database\Seeders;

use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use App\Models\Landing;

class LandingSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        //
        Landing::factory()->count(3)->create();

        Landing::create([
            'title' => 'First Book Title',
            'rating' => 4,
            'author' => 'First Book Author',
            'created_at' => now(),
            'updated_at' => now(),
        ]);
    }
}
```



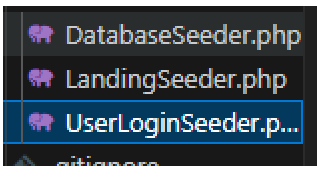


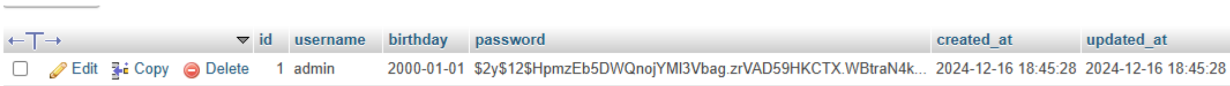
□ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None
Extra options

←T→				id	title	rating	author	created_at	updated_at
□	Edit	Copy	Delete	1	Quis molestiae eum aut porro aut rerum.	4	Suscipit culpa vitae quis voluptas ipsa qui iure q...	2024-12-16 16:01:58	2024-12-16 16:01:58
□	Edit	Copy	Delete	2	Facere qui fuga exercitationem est id voluptates r...	2	Quia voluptatem laboriosam aut voluptatem officia ...	2024-12-16 16:01:58	2024-12-16 16:01:58
□	Edit	Copy	Delete	3	Soluta ut distinctio quam consequatur.	1	Animi et id tempore excepturi vel omnis.	2024-12-16 16:01:58	2024-12-16 16:01:58
□	Edit	Copy	Delete	4	First Book Title	4	First Book Author	2024-12-16 16:01:58	2024-12-16 16:01:58



```php
use App\Models\UserLogin;

class UserLoginSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        //
        UserLogin::create([
            'username' => 'admin',
            'birthday' => '2000-01-01',
            'password' => bcrypt('admin123'),
            'created_at' => now(),
            'updated_at' => now(),
        ]);
    }
}
```

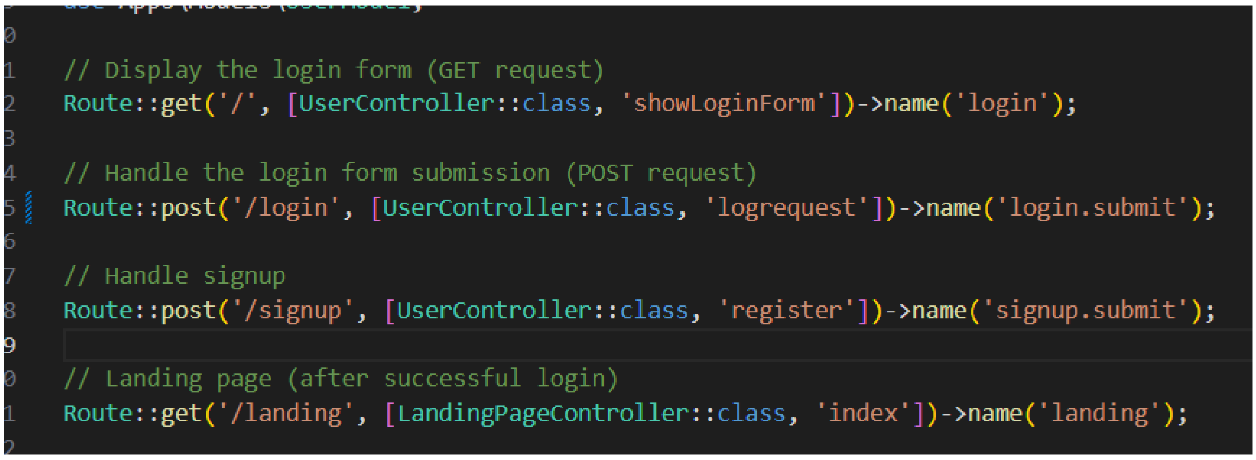




←T→				id	username	birthday	password	created_at	updated_at
□	Edit	Copy	Delete	1	admin	2000-01-01	$2y$12$HpmzEb5DWQnojYMI3Vbag.zrVAD59HKCTX.WBtraN4k...	2024-12-16 18:45:28	2024-12-16 18:45:28

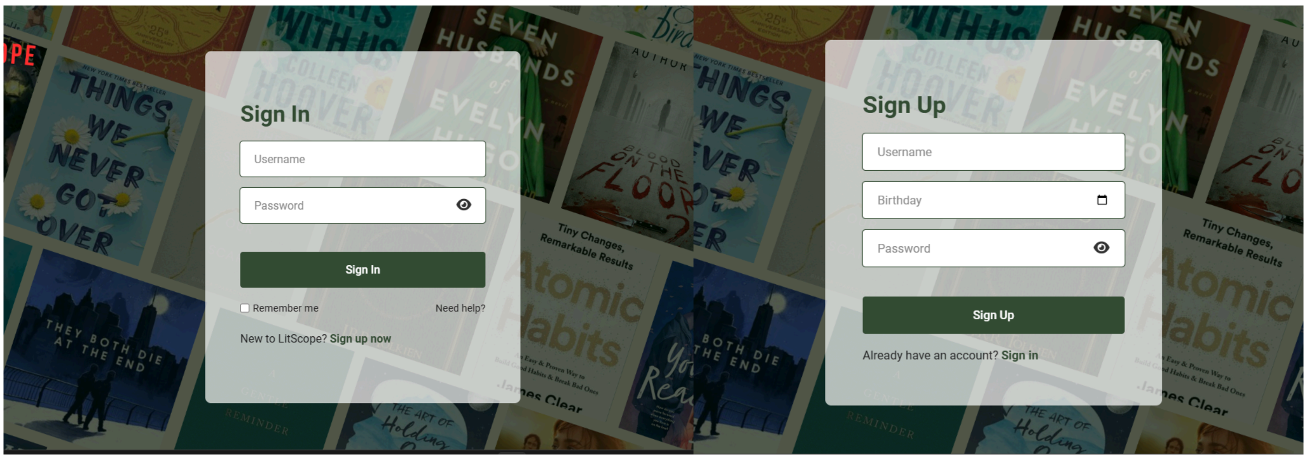
Using ‘php artisan db:class –class=SeederName’, it will automatically populate the table with the initial data: refer to the redbox.

Define the routes and register the controllers.



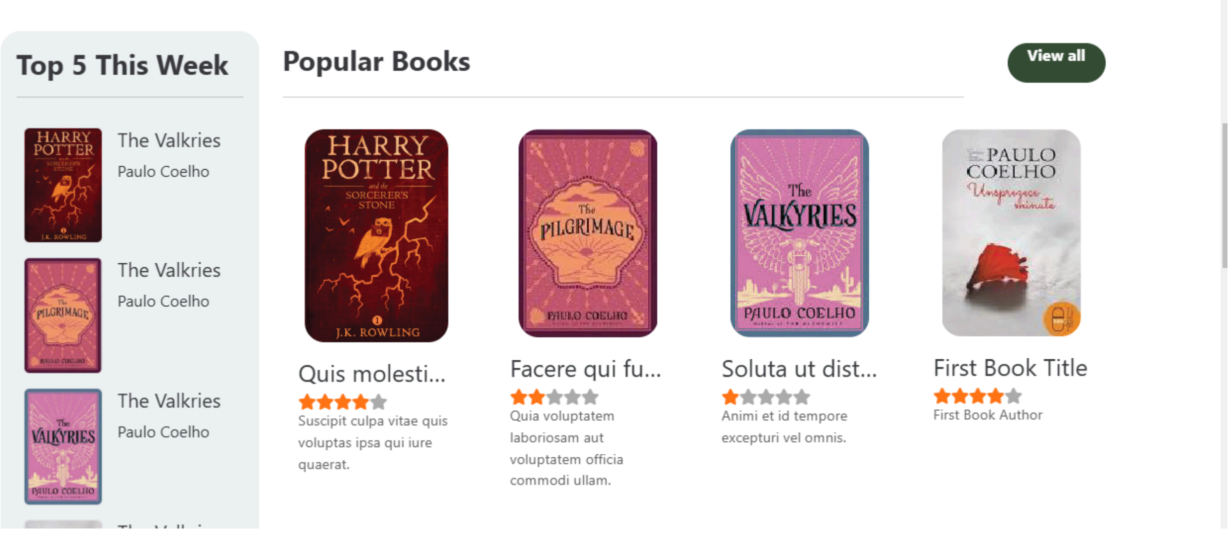
```
use App\Models\User;

// Display the login form (GET request)
Route::get('/', [UserController::class, 'showLoginForm'])->name('login');

// Handle the login form submission (POST request)
Route::post('/login', [UserController::class, 'logrequest'])->name('login.submit');

// Handle signup
Route::post('/signup', [UserController::class, 'register'])->name('signup.submit');

// Landing page (after successful login)
Route::get('/landing', [LandingPageController::class, 'index'])->name('landing');
```

The ‘/’ and ‘signup’ with a Controller takes a ‘showLoginForm’ and ‘register’ functions to display the login form and signup form.



Once they enter their username and password, laravel checks if the credentials is stored in the database, if yes, the user will be directed to ‘/landing’ which corresponds to the homepage.

Homepage:



.

If a new user creates an account, it will be stored in the database, and they will be directed to ‘/’ to display the login form, and the process will begin again.