**Autószerelő Műhely weboldala Laravel keretrendszerrel**

A feladat leírásában megkapsz minden segítséget, de azt Neked kell megtalálni, hogy azt hol kell felhasználni. A feladat célja, hogy gyakorold a Laravel keretrendszer szerkezetének használatát.

**Első lépések:**

Hozz létre egy új Laravel projektet, amely az "Autószerelő Műhely" témát dolgozza fel. Ellenőrizd, hogy a projekt működik-e a böngészőben.

Módosítsd a kezdőoldalt úgy, hogy megjelenítse a következő üzenetet: "Üdvözlünk az Autószerelő Műhely nyilvántartási rendszerében!"

Route::get('/', function () { return "<h1>Üdvözlünk az Autószerelő Műhely nyilvántartási rendszerében!</h1>"; });

**Autólistázás:**

Hozz létre egy nézetet és egy CarController vezérlőt, amely egy listát jelenít meg a műhelybe érkező autókról.

php artisan make:controller CarController

Az autók adatai legyenek statikusan definiálva a vezérlőben (pl. márka, typel, rendszám).

A nézetben egy táblázat formájában jelenítsd meg ezeket az adatokat.

CarController:

public function index() {

$cars = [

['brand' => 'Suzuki', 'type' => 'Swift', 'plate' => 'ABC-123'],

['brand' => 'Ford', 'type' => 'Focus', 'plate' => 'XYZ-789'],

['brand' => 'BMW', 'type' => '320i', 'plate' => 'DEF-456'],

];

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

}

cars/cars.blade.php

Nézet:

<table>

<thead>

<tr>

<th>Márka</th>

<th>Typel</th>

<th>Rendszám</th>

</tr>

</thead>

<tbody>

@foreach ($cars as $car)

<tr>

<td>{{ $car['brand'] }}</td>

<td>{{ $car['type'] }}</td>

<td>{{ $car['plate'] }}</td>

</tr>

@endforeach

</tbody>

</table>

Hozz létre egy dinamikus útvonalat, amely egy adott autó részleteit jeleníti meg a rendszám alapján.

use App\Http\Controllers\CarController;

Route::get('/cars/{plate}', [CarController::class, 'show']);

Kontrollerben hozd létre a show metódust:

ne használd a type-t :D ütközés lesz!!! át kellett volna írni type-ra

public function show($plate) {

$cars = [

'ABC-123' => ['brand' => 'Suzuki', 'type' => 'Swift', 'plate' => 'ABC-123'],

'XYZ-789' => ['brand' => 'Ford', 'type' => 'Focus', 'plate' => 'XYZ-789'],

'DEF-456' => ['brand' => 'BMW', 'type' => '320i', 'plate' => 'DEF-456'],

];

if (!array\_key\_exists($plate, $cars)) {

return "Nincs ilyen autó a rendszerben.";

}

return view('cars.show', ['car' => $cars[$plate]]);

}

cars/show.blade.php fájlban jelenítsd meg az autó adatait

<h1>{{ $car['brand'] }} {{ $car['type'] }}</h1>

<p>Rendszám: {{ $car['plate'] }}</p>

Hozz létre egy cars táblát az adatbázisban. A tábla tartalmazza az autók adatait: rendszám, márka, typel, évjárat, és hiba leírása.

php artisan make:migration create\_cars\_table

Egészítsd ki a migrációs fájlt:

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

$table->id();

$table->string('plate')->unique();

$table->string('brand');

$table->string('type');

$table->year('year');

$table->text('issue');

$table->timestamps();

});

Készíts funkciókat az autók adatainak hozzáadására, szerkesztésére és törlésére.

cars/create.blade.php

<form action="/cars" method="POST">

@csrf

<label for="plate">Rendszám:</label>

<input type="text" name="plate" id="plate" required>

<label for="brand">Márka:</label>

<input type="text" name="brand" id="brand" required>

<label for="type">Modell:</label>

<input type="text" name="type" id="type" required>

<label for="year">Évjárat:</label>

<input type="number" name="year" id="year" required>

<label for="issue">Hiba leírása:</label>

<textarea name="issue" id="issue" required></textarea>

<button type="submit">Hozzáadás</button>

</form>

A vezérlőben hozz létre egy store() metódust, amely az adatokat elmenti az adatbázisba.

public function store(Request $request) {

// Adatok validálása

$request->validate([

'plate' => 'required|unique:cars|max:10',

'brand' => 'required|string|max:50',

'type' => 'required|string|max:50',

'year' => 'required|integer|min:1900|max:' . date('Y'),

'issue' => 'required|string',

]);

// Adatok mentése

$car = new Car();

$car->plate = $request->plate;

$car->brand = $request->brand;

$car->type = $request->type;

$car->year = $request->year;

$car->issue = $request->issue;

$car->save();

// Visszairányítás

return redirect('/cars')->with('success', 'Az autó adatai sikeresen elmentve!');

}

php artisan make:model Car

import class vagy

use App\Models\Car;

Add hozzá a stíluslapot:

public/css/style.css

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid #ddd;

padding: 8px;

}

form {

display: flex;

flex-direction: column;

gap: 10px;

}

Linkeld be a CSS fájlt a layouts/main.blade.php sablonodban:

<link rel="stylesheet" href="{{ asset('css/style.css') }}">

Módosítás: Készíts egy szerkesztő űrlapot, amely előre kitölti a meglévő adatokat.

A vezérlőben hozz létre egy update() metódust, amely frissíti az adatokat.

Szükség lesz a megfelelő útvonalakra:

Route::get('/cars/{id}/edit', [CarController::class, 'edit'])->name('cars.edit');

Route::put('/cars/{id}', [CarController::class, 'update'])->name('cars.update');

A vezérlőben hozz létre egy edit() metódust, amely betölti a meglévő adatokat, és továbbítja őket a nézetnek:

use App\Types\Car;

public function edit($id) {

$car = Car::findOrFail($id); // Az ID alapján betöltjük az adatokat

return view('cars.edit', compact('car')); // Átadjuk a nézetnek

}

cars/edit.blade.php –ben hozd létre a módosítás űrlapját:

<form action="{{ route('cars.update', $car->id) }}" method="POST">

@csrf

@method('PUT')

<label for="plate">Rendszám:</label>

<input type="text" name="plate" id="plate" value="{{ $car->plate }}" required>

<label for="brand">Márka:</label>

<input type="text" name="brand" id="brand" value="{{ $car->brand }}" required>

<label for="type">Typel:</label>

<input type="text" name="type" id="type" value="{{ $car->type }}" required>

<label for="year">Évjárat:</label>

<input type="number" name="year" id="year" value="{{ $car->year }}" required>

<label for="issue">Hiba leírása:</label>

<textarea name="issue" id="issue" required>{{ $car->issue }}</textarea>

<button type="submit">Mentés</button>

</form>

A vezérlőben hozz létre egy update() metódust, amely fogadja a frissített adatokat, validálja azokat, és elmenti az adatbázisba:

public function update(Request $request, $id) {

// Adatok validálása

$request->validate([

'plate' => 'required|max:10|unique:cars,plate,' . $id,

'brand' => 'required|string|max:50',

'type' => 'required|string|max:50',

'year' => 'required|integer|min:1900|max:' . date('Y'),

'issue' => 'required|string',

]);

// Adatok frissítése

$car = Car::findOrFail($id);

$car->plate = $request->plate;

$car->brand = $request->brand;

$car->type = $request->type;

$car->year = $request->year;

$car->issue = $request->issue;

$car->save();

// Visszairányítás és sikerüzenet

return redirect('/cars')->with('success', 'Az autó adatai sikeresen frissítve!');

}

Tesztelés:

Navigálj a böngészőben a /cars/{id}/edit útvonalra.

Töltsd ki az űrlapot új adatokkal, és küldd el.

Ellenőrizd, hogy az adatok frissültek-e az adatbázisban, és hogy megjelenik-e a sikerüzenet.

**Idáig sikeresen el kellett jutnod. ☺**

Tegyük lehetővé a felvitelt:

//felvitel

Route::get('/cars/create', [CarController::class, 'create'])->name('cars.create');

//mentés

Route::post('/cars', [CarController::class, 'store'])->name('cars.store');

Egészítsd ki a vezérlőt, ahol kell ott pedig módosítsd a metódusokat.

public function index()

{

$cars = Car::all(); // Összes autó lekérdezése az adatbázisból

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

}

public function create()

{

return view('cars.create'); // Az űrlapot tartalmazó nézet

}

public function show($plate)

{

$car = Car::where('plate', $plate)->first(); // Autó lekérdezése rendszám alapján

if (!$car) {

return "Nincs ilyen autó a rendszerben.";

}

return view('cars.show', compact('car'));

}

**Adjuk hozzá a szükséges útvonalakat:**

Route::get('/cars', [CarController::class, 'index'])->name('cars.index');

Route::get('/cars/{plate}', [CarController::class, 'show'])->name('cars.show');

**Miért jó ha van egy layout.blade.php fájlod?**

A @yield('content') szekció lehetővé teszi, hogy az egyes nézetek (index.blade.php, show.blade.php, stb.) saját tartalmat illesszenek be ide.

A header és footer minden oldalon megjelenik, így egységes lesz a design.

A resource/view mappa alá hozd létre a layout.blade.php fájlt a következő tartalommal:

<!DOCTYPE html>

<html lang="hu">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Autószerelő Műhely</title>

</head>

<body>

<header>

<h1>Autószerelő Műhely</h1>

<nav>

<ul>

<li><a href="{{ route('cars.index') }}">Autók listája</a></li>

<li><a href="{{ route('cars.create') }}">Új autó felvétele</a></li>

</ul>

</nav>

</header>

<main>

@yield('content')

</main>

<footer>

<p>&copy; {{ date('Y') }} Autószerelő Műhely</p>

</footer>

</body>

</html>

**Készítsd el a cars mappában a show.blade.php fájlt:**

@extends('layout')

@section('content')

<h1>Autó adatai</h1>

<p><strong>Márka:</strong> {{ $car->brand }}</p>

<p><strong>Típus:</strong> {{ $car->type }}</p>

<p><strong>Rendszám:</strong> {{ $car->plate }}</p>

<p><strong>Év:</strong> {{ $car->year }}</p>

<p><strong>Probléma:</strong> {{ $car->issue }}</p>

<a href="{{ route('cars.index') }}">Vissza a listához</a>

@endsection

A stílulap hivatkozást másold át a layout.blade.php oldalra.