**Курс «Создание web–приложений, исполняемых на стороне сервера при помощи языка программирования PHP и технологии AJAX»**

**Тема: Laravel.**

**Домашняя работа №4**

Выполнил студент гр. ВПУ-911: Мулин Николай Сергеевич

IDE: Microsoft Visual Studio Code

Ссылка на проект в GitHub: <https://github.com/Nickomu92/hw4_php>

**Условие задания**

1. *Создать сайт для предоставления информации о знаках зодиаков. Например, пользователь вводит дату рождения и получает информацию о своем знаке зодиака (классический календарь, календарь друидов, восточный календарь и т.д.).*

**Создание проекта:**

1. *Переходим в папку, где хотим создать проект и в командной строке выполняем команду:*

**composer create-project laravel/laravel HW\_PHP\_4\_Mulina\_Nikolaya**

1. *Создаем контроллер (мы создаем RESTful Controller в котором уже будут созданы заготовки методов для выполнения стандартных действий) командой:*

**php artisan make:controller HoroscopeController -resource**

1. *Подключаем модуль Form (для работы с формами) командой:*

**composer require "laravelcollective/html"**

**Проект состоит из следующих файлов**

*(указаны только скриптовые файлы, а остальные -* [*тут*](https://github.com/Nickomu92/hw4_php)*):*

1. ***Controllers:***

* ***HoroscopeController.php***

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

// Подключаем модуль (пакет) для работы со временем и датой

use Carbon\Carbon;

// Класс для представления сущности пользователя

class User

{

// Поле для хранения даты рождения пользователя

protected $date;

public function \_\_construct($date)

{

$this->date = $date;

}

// Метод для получения года рождения пользователя

public function getYear()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

return $date['year'];

}

// Метод для получения месяца рождения пользователя

public function getMonth()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

return $date['month'];

}

// Метод для получения дня рождения пользователя

public function getDay()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

return $date['day'];

}

// Метод для получение информации по классическому календарю

public function getClassicZodiac()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

$day = $date['day'];

$month = $date['month'];

if(($day >= 21 && $day <= 31 && $month == 3) || ($day <= 20 && $month == 4))

return 'Овен';

else if(($day >= 21 && $day <= 30 && $month == 4) || ($day <= 21 && $month == 5))

return 'Телец';

else if(($day >= 22 && $day <= 31 && $month == 5) || ($day <= 21 && $month == 6))

return 'Близнецы';

else if(($day >= 23 && $day <= 30 && $month == 6) || ($day <= 22 && $month == 7))

return 'Рак';

else if(($day >= 23 && $day <= 31 && $month == 7) || ($day <= 21 && $month == 8))

return 'Лев';

else if(($day >= 22 && $day <= 31 && $month == 8) || ($day <= 23 && $month == 9))

return 'Дева';

else if(($day >= 24 && $day <= 30 && $month == 9) || ($day <= 23 && $month == 10))

return 'Весы';

else if(($day >= 24 && $day <= 31 && $month == 10) || ($day <= 22 && $month == 11))

return 'Скорпион';

else if(($day >= 23 && $day <= 30 && $month == 11) || ($day <= 22 && $month == 12))

return 'Стрелец';

else if(($day >= 23 && $day <= 31 && $month == 12) || ($day <= 20 && $month == 1))

return 'Козерог';

else if(($day >= 21 && $day <= 31 && $month == 1) || ($day <= 19 && $month == 2))

return 'Водолей';

else if(($day >= 20 && $day <= 29 && $month == 2) || ($day <= 20 && $month == 3))

return 'Рыбы';

}

// Метод для получения возраста пользователя

public function getAge()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

$year = $date['year'];

$month = $date['month'];

$day = $date['day'];

// Модуль Carbon для работы со временем и датой

$date = Carbon::now();

$date->year($year)->month($month)->day($day);

$dateNow = Carbon::now();

$resultYear = $dateNow->diffInYears($date);

if($resultYear % 10 == 1)

return $resultYear . ' год';

else if($resultYear % 10 >= 2 && $resultYear % 10 <= 4)

return $resultYear . ' года';

else

return $resultYear . ' лет';

}

// Метод для получение информации по китайскому календарю

public function getChineseZodiac()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

// 12 - количество знаков зодиака

$chineseZodiac = $date['year'] % 12;

switch(abs($chineseZodiac))

{

case 0:

return 'Обезьяна';

case 1:

return 'Петух';

case 2:

return 'Собака';

case 3:

return 'Свинья';

case 4:

return 'Крыса';

case 5:

return 'Бык';

case 6:

return 'Тигр';

case 7:

return 'Кролик';

case 8:

return 'Дракон';

case 9:

return 'Змея';

case 10:

return 'Лошадь';

case 11:

return 'Коза';

}

}

// Метод для получение информации календарю друидов

public function getDruidicZodiac()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

$day = $date['day'];

$month = $date['month'];

if(($day > 22 && $day <= 31 && $month == 12) || ($day <= 1 && $month == 1) || ($day >= 25 && $day <= 30 && $month == 6) || ($day <= 4 && $month == 7))

return 'Яблоня';

else if(($day >= 2 && $day <= 11 && $month == 1) || ($day >= 5 && $day <= 14 && $month == 7))

return 'Пихта';

else if(($day >= 12 && $day <= 24 && $month == 1) || ($day >= 15 && $day <= 25 && $month == 7))

return 'Вяз';

else if(($day >= 25 && $day <= 31 && $month == 1) || ($day <= 3 && $month == 2) || ($day >= 26 && $day <= 31 && $month == 7) || ($day <= 4 && $month == 8))

return 'Кипарис';

else if(($day >= 4 && $day <= 8 && $month == 2) || ($day >= 5 && $day <= 13 && $month == 8))

return 'Тополь';

else if(($day >= 9 && $day <= 18 && $month == 2) || ($day >= 14 && $day <= 23 && $month == 8))

return 'Кедр';

else if(($day >= 19 && $day <= 29 && $month == 2) || ($day >= 24 && $day <= 31 && $month == 8) || ($day <= 2 && $month == 9))

return 'Сосна';

else if(($day >= 1 && $day <= 10 && $month == 3) || ($day >= 3 && $day <= 12 && $month == 9))

return 'Ива';

else if(($day >= 11 && $day < 20 && $month == 3) || ($day >= 13 && $day < 23 && $month == 9))

return 'Липа';

else if(($day >= 22 && $day <= 31 && $month == 3) || ($day >= 24 && $day <= 31 && $month == 9) || ($day <= 3 && $month == 9))

return 'Орешник';

else if(($day >= 1 && $day <= 10 && $month == 4) || ($day >= 4 && $day <= 13 && $month == 10))

return 'Рябина';

else if(($day >= 11 && $day <= 20 && $month == 4) || ($day >= 14 && $day <= 23 && $month == 10))

return 'Клен';

else if(($day >= 21 && $day <= 31 && $month == 4) || ($day >= 24 && $day <= 31 && $month == 10) || ($day <= 2 && $month == 11))

return 'Грецкий орех';

else if(($day >= 1 && $day <= 14 && $month == 5) || ($day >= 3 && $day <= 11 && $month == 11))

return 'Жасмин';

else if(($day >= 15 && $day <= 24 && $month == 5) || ($day >= 12 && $day <= 21 && $month == 11))

return 'Каштан';

else if(($day >= 25 && $day <= 31 && $month == 5) || ($day <= 3 && $month == 6) || ($day >= 22 && $day <= 30 && $month == 11) || ($day <= 1 && $month == 12))

return 'Ясень';

else if(($day >= 4 && $day <= 13 && $month == 6) || ($day >= 2 && $day <= 11 && $month == 12))

return 'Граб';

else if(($day >= 14 && $day < 24 && $month == 6) || ($day >= 12 && $day < 21 && $month == 12))

return 'Инжир';

else if($day == 20 && $month == 3)

return 'Дуб';

else if($day == 24 && $month == 6)

return 'Береза';

else if($day == 23 && $month == 9)

return 'Олива';

else if($day >= 21 && $day <= 22 && $month == 9)

return 'Бук';

}

// Метод для получение информации по зороатрийскому календарю

public function getZoroastrianZodiac()

{

// Парсим дату рождения пользователя

$date = date\_parse($this->date);

// 32 - количество знаков зодиака

$zoroastrianZodiac = $date['year'] % 32;

switch(abs($zoroastrianZodiac))

{

case 0:

return 'Барсук';

case 1:

return 'Верблюд';

case 2:

return 'Еж';

case 3:

return 'Лань';

case 4:

return 'Слон';

case 5:

return 'Конь';

case 6:

return 'Гепард';

case 7:

return 'Павлин';

case 8:

return 'Лебедь';

case 9:

return 'Рысь';

case 10:

return 'Осел';

case 11:

return 'Белый Медведь';

case 12:

return 'Орел';

case 13:

return 'Лиса';

case 14:

return 'Дельфин';

case 15:

return 'Вепрь';

case 16:

return 'Сова';

case 17:

return 'Сокол';

case 18:

return 'Олень';

case 19:

return 'Баран';

case 20:

return 'Мангуст';

case 21:

return 'Волк';

case 22:

return 'Аист';

case 23:

return 'Паук';

case 24:

return 'Змея';

case 25:

return 'Бобер';

case 26:

return 'Черепаха';

case 27:

return 'Сорока';

case 28:

return 'Белка';

case 29:

return 'Ворон';

case 30:

return 'Петух';

case 31:

return 'Тур';

}

}

}

// RESTful контроллер с созданными по умолчанию методами

class HoroscopeController extends Controller

{

public function index()

{

// Модуль Carbon для работы со временем и датой

$user = new User(Carbon::now());

$info = 'Пример (сегодняшня дата)';

return view('page', ['user' => $user, 'info' => $info]);

}

public function create()

{

//

}

public function store(Request $request)

{

$user = new User($request->date);

return view('page', ['user' => $user]);

}

public function show($id)

{

//

}

public function edit($id)

{

//

}

public function update(Request $request, $id)

{

//

}

public function destroy($id)

{

//

}

}

1. ***Views:***

* ***content.blade.php***

<!-- Использование модуля Form -->

{{ Form::model($user, ['url' => route('horoscope.store'), 'class' => 'container-fluid m-2']) }}

{{ Form::label('date', 'Укажите дату Вашего рождения: ', ['class' => 'h4 m-3 text-primary'])}}

{{ Form::date('date', \Carbon\Carbon::now()) }}

{{ Form::submit('Получить данные', ['class' => 'btn btn-success m-l-2']) }}

{{ Form::close() }}

@if(isset($user))

@if(isset($info))

<div class="container-fluid text-center h4 text-danger">

{{$info}}

</div>

@endif

<div class="container-fluid">

<div class="alert alert-primary" role="alert">

Ваша дата рождения: {{ $user->getDay() . '/' . $user->getMonth() . '/' . $user->getYear() }} г.

</div>

<div class="alert alert-secondary" role="alert">

Возраст (полных лет): {{$user->getAge() }}

</div>

<div class="alert alert-success" role="alert">

Знак зодиака: {{ $user->getClassicZodiac() }}

</div>

<div class="alert alert-danger" role="alert">

По восточному гороскопу: {{ $user->getChineseZodiac() }}

</div>

<div class="alert alert-warning" role="alert">

По зороастрийский гороскопу: {{ $user->getZoroastrianZodiac() }}

</div>

<div class="alert alert-dark" role="alert">

По календарю друидов: {{ $user->getDruidicZodiac() }}

</div>

</div>

@endif

* ***footer.blade.php***

<!-- Нижняя часть сайта -->

<div class="conteiner-fluid bg-dark text-light text-center h4 p-2">2021 NikoMu Horoscope&copy;</div>

</body>

</html>

* ***header.blade.php***

<!-- Верхняя часть сайта -->

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

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

<script src="{{asset('js/bootstrap.min.js')}}"></script>

<title>NikoMu Horoscope</title>

</head>

<body>

<nav class="navbar navbar-dark bg-dark">

<div class="container-fluid">

<span class="text-danger h3">Niko<span class="text-warning h3">Mu</span><span class="text-success h4"> Horoscope</span></span>

</div>

</nav>

* ***page.blade.php***

@extends('footer')

@extends('content')

@extends('header')

1. ***Routes***

* ***web.php***

<?php

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\HoroscopeController;

Route::get('/', function () {

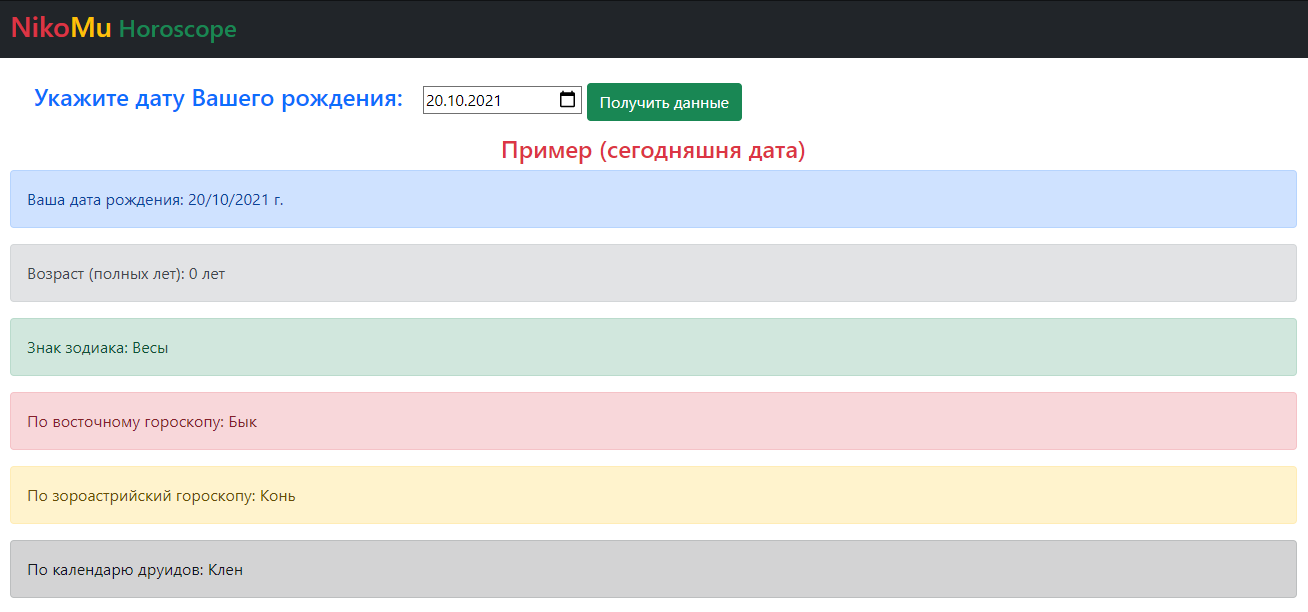
return view('welcome');

});

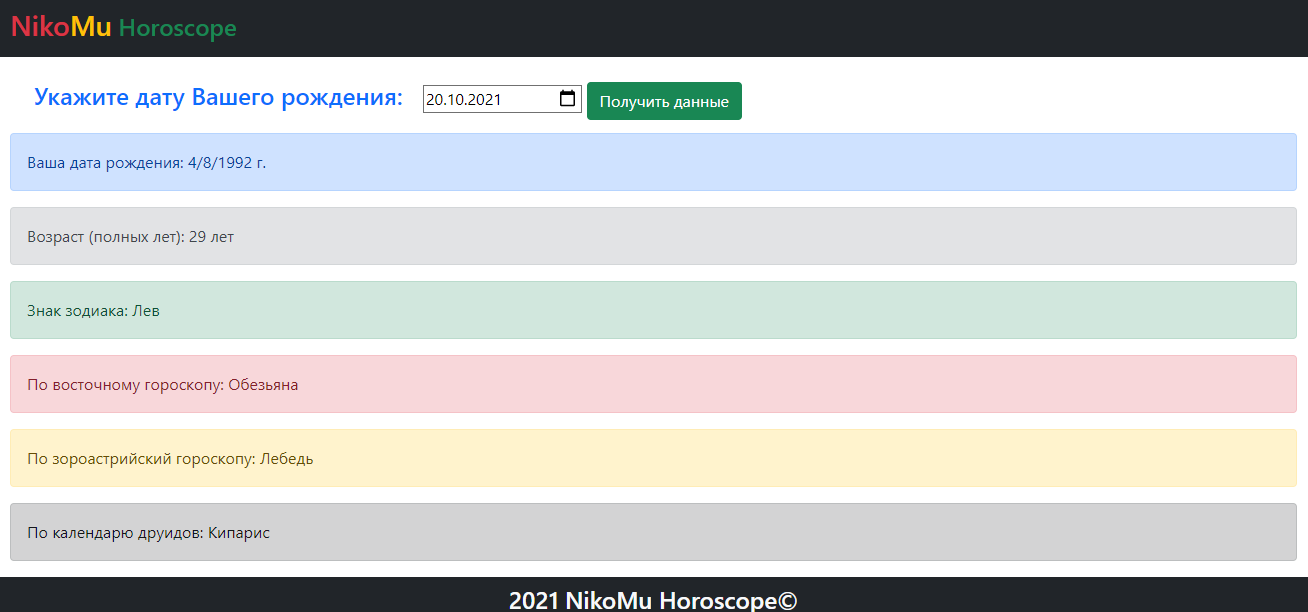
// Указываем маршрутизацию для всех методов RESTful контроллера - HoroscopeController

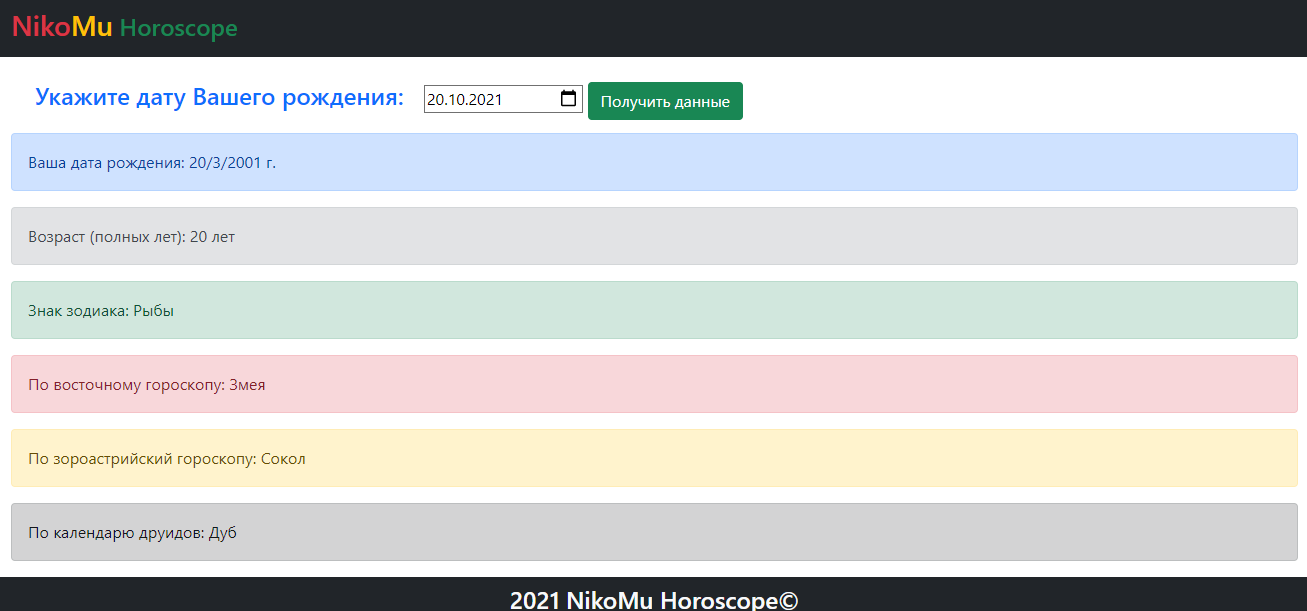
Route::resource('/horoscope', HoroscopeController::class);

**Результат работы:**



*Рис.1 - Начальный вид страницы отработал метод index() контроллера HoroscopeController*





*Рис.2-3 - Вид страницы после отправки формы - отработал метод store() контроллера HoroscopeController*