Урок 2. Контроллеры, экшены и роутинг

**Цели:**  
  
Вы научитесь:

— устанавливать Laravel;  
— создавать контроллер, возвращающий JSON;  
— создавать контроллер для обработки формы;  
— создавать контроллер, возвращающий шаблон.

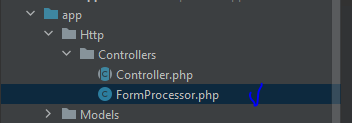
**Что нужно сделать:**

Вам предстоит установить фреймворк Laravel и создать контроллер, содержащий экшены для вывода и обработки формы.  
  
1. Установите Laravel с помощью composer, выполнив команду composer **create-project laravel/laravel** ***<имя проекта>***. В поле <имя проекта> впишите имя вашего проекта. Этому имени будет соответствовать имя папки, в которую вы поместите проект.

Выполнено в дз 1  
  
2. Создайте контроллер для вывода формы на страницу и её обработки. В командную строку введите команду **php artisan make:controller FormProcessor**.

Команда работает корректно

3. После выполнения команды убедитесь, что контроллер создан, — соответствующий файл должен появиться в папке app/Http/Controllers.



4. Внутри контроллера опишите метод index: он должен выводить в браузер форму для заполнения.  
— Опишите форму в виде шаблона blade.  
— Внутри формы должны быть поля для ввода имени, фамилии и email пользователя.  
— Форма отправляется методом POST.  
— Параметр action пока оставьте пустым (в пояснении ниже action уже не пустой – забыла скрин сделать вовремя)  
— Не забудьте про CSRF.

1. В контроллере пишем метод index, который будет просто выводить форму на страницу
2. Сразу прописываем роут на get запрос. Папка \routes\web.php

Route::get('/userform', [\App\Http\Controllers\FormProcessor::class, 'index']) -> name('userform.index');

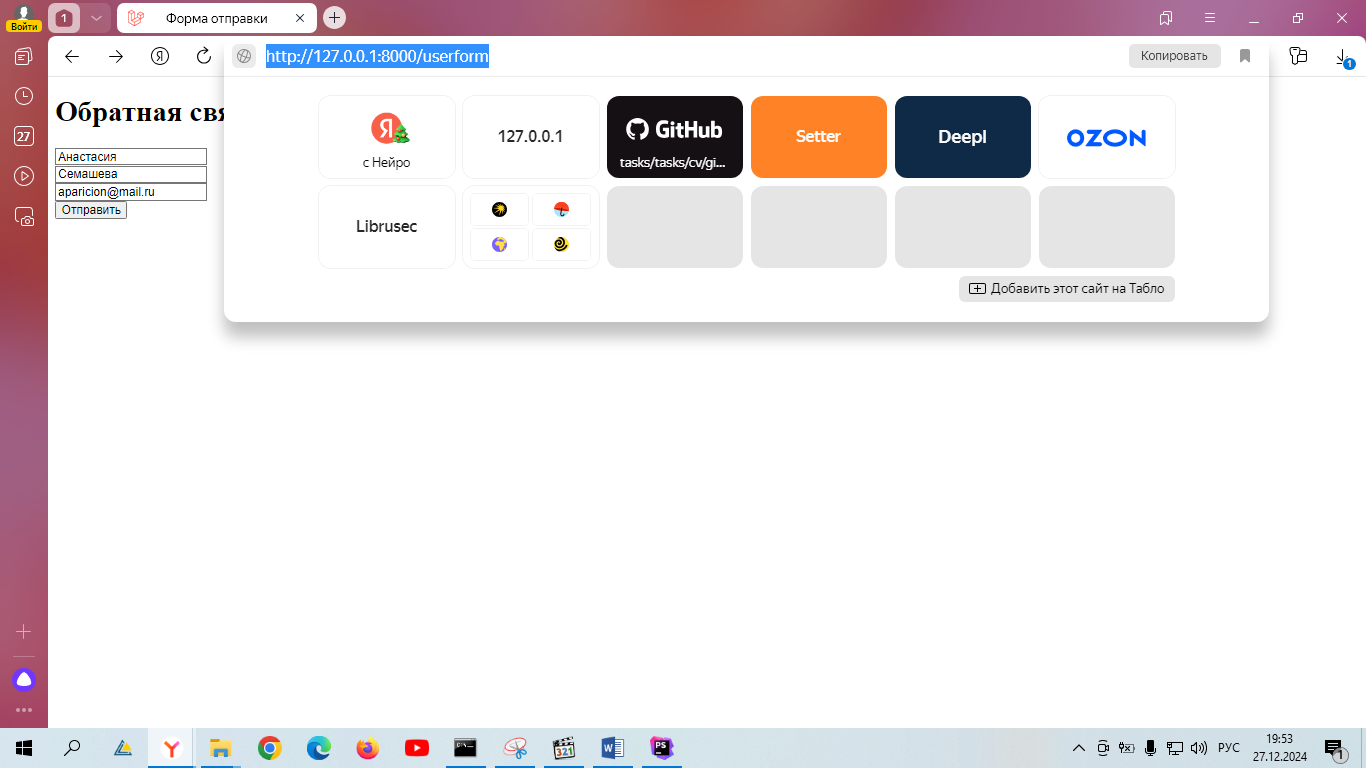
1. Создаем шаблон формы связи в директории resources/views/invdex.blade.php
2. @extends('layouts.app')
3. @section('content')
4. <h1>Обратная связь</h1>
5. @if (session('success'))
6. <div class="alert alert-success" role="alert">
7. {{ session('success') }}
8. </div>
9. @endif
10. @if ($errors->any())
11. <div class="alert alert-danger" role="alert">
12. <ul>
13. @foreach ($errors->all() as $error)
14. <li>{{ $error }}</li>
15. @endforeach
16. </ul>
17. </div>
18. @endif
19. <form method="post" action="{{ route('feedback.send') }}">
20. @csrf
21. <div class="form-group">
22. <input type="text" class="form-control" name="name" placeholder="Имя, фамилия"
23. required maxlength="100" value="{{ old('name') ?? '' }}">
24. </div>
25. <div class="form-group">
26. <input type="email" class="form-control" name="email" placeholder="Адрес почты"
27. required maxlength="100" value="{{ old('email') ?? '' }}">
28. </div>
29. <div class="form-group">
30. <textarea class="form-control" name="message" placeholder="Ваше сообщение"
31. required maxlength="500" rows="3">{{ old('message') ?? '' }}</textarea>
32. </div>
33. <div class="form-group">
34. <button type="submit" class="btn btn-primary">Отправить</button>
35. </div>
36. </form>
37. @endsection

@extends('layouts.app') – шаблон в директории resources/views/layouts/app.blade.php

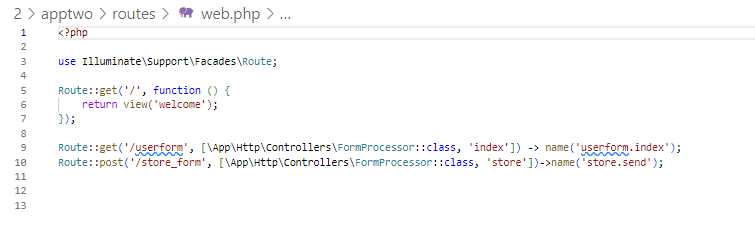
Собственно основа сайта

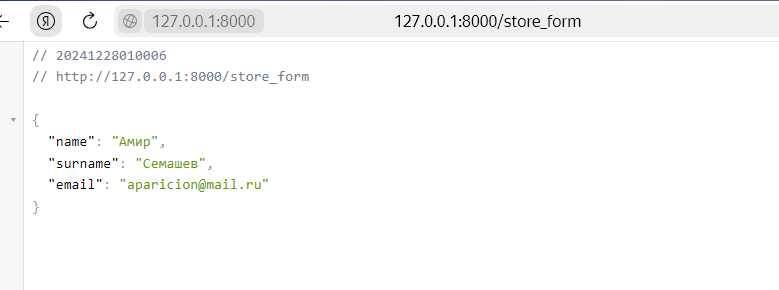
6. Запустите встроенный сервер Laravel командой php artisan serve --port=8080 и убедитесь, что форма выводится по адресу <http://localhost:8080/userform>.

Скрин ниже

  
  
7. В контроллере **FormProcessor** создайте метод **store** для обработки формы. Этот метод должен принимать поля формы и отправлять ответ в виде JSON-объекта, содержащего значения полей формы (имя, фамилия, email).

  
  
8. Внутри файла /routes/web.php опишите новый роут (метод POST), который будет вызывать метод **store** контроллера **FormProcessor** по url **/store\_form**.

  
9. Отредактируйте поле action формы в шаблоне и укажите адрес /store\_form.  
  
10. Откройте форму в браузере по адресу <http://localhost:8080/userform>, заполните её и попробуйте отправить на сервер, нажав кнопку Submit. Если всё сделано правильно, вы увидите в браузере объект JSON.

  
11. Создайте новый шаблон blade для приветствия пользователя (например: «Привет, <имя>!»).

  
12. Измените метод store контроллера FormProcessor таким образом, чтобы вместо JSON он возвращал шаблон, заполненный данными пользователя.

  
  
13. Сделайте коммит своих изменений с помощью git и отправьте push в репозиторий.

**Список файлов, которые нужны были и верно созданы:**

1. app/Http/Controllers/FormProcessor.php

Добавлены два метода index(вывод формы на страницу) и store(формирование данных из введенной в форму информации)

1. \routes\web.php роут на get и post запросы
2. resources/views/invdex.blade.php

Создание шаблона формы для ввода информации

1. resources/views/layouts/app.blade.php

Создание основы сайта

1. resources/views/hello.blade.php

Формирование шаблона для вывода данных, введенных Пользователем в форму

**Советы и рекомендации**  
Старайтесь писать в commit message то, что отражает смысл вашего кода.

**Критерии оценки работы:**

**Принято: \*\*  
— выполнены все пункты задания;  
— в работе используются указанные инструменты и соблюдены условия;  
— код корректно отформатирован по стандартам программирования на PHP;  
— скрипт запускается, выводит различные данные на экран, не вызывает ошибок.  
  
\*\*На доработку:**  
— выполнены не все обязательные пункты задания;  
— задание выполнено с ошибками.  
  
**Как отправить работу на проверку**  
Отправьте коммит, содержащий код задания, на ветку master в вашем репозитории и пришлите его URL (URL Merge Request’а) через форму. Репозиторий должен быть public.