УДК 004.4’24

ИСПОЛЬЗОВАНИЕ ФРЕЙМВОРКА LARAVEL 5.0 ДЛЯ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЙ

***Брусов Алексей Сергеевич**, студент,*
***Тарасов Сергей Олегович**, студент,*
Юго-Западный государственный университет, Россия

В статье рассматривается использование фреймворка Laravel 5.0 для разработки web-приложения «Инфраструктурный сервис поддержки предпринимателей». Описаны основные используемые методы взаимодействия клиента с сервером и сервера с базой данных на примере сценария действия просмотра страницы.

Ключевые слова: фреймворк, клиент, сервер, база данных, запрос, модель, веб-приложение, фильтр, контроллер.

Проблемы, возникающие у любого веб-разработчика связаны с затратами времени на алгоритмы аутентификации пользователей, обработку и фильтрацию запросов, настройку взаимодействия с базой данных, а также множества других стандартных для веб-приложения возможностей.

С целью упрощения создания архитектуры программных продуктов были разработаны фреймворки (англ. *framework* — каркас, структура). Так вот, фреймворк – это платформа, определяющая структуру программного продукта. Соответственно веб-фреймворк – это платформа, в большинстве случаев использующая язык программирования php, определяющая структуру веб-приложения. В настоящее время существует огромное количество таких платформ, но в данной статье рассмотрена одна из самых простых и многофункциональных – Laravel.

PHP-фреймворк Laravel как и большинство современных фреймворков бесплатный и распространяется с открытым исходным кодом. Он предназначен для разработки веб-приложений с использованием архитектурной модели Модель-Представление-Контроллер (Model View Controller).

Каждое веб-приложение имеет определенные аппаратные компоненты (узлы), физическое развертывание которых моделирует диаграмма развертывания [1]. Пример такой диаграммы схематично представлен на рисунке 1.

Рассмотренная структура веб-приложения, состоит из двух главных компонентов – веб-клиента и веб-сервера. Веб-клиентом в любом таком продукте выступает интернет обозреватель, браузер. Серверная часть может быть различной. В данном примере сервер развернут на операционной системе Centos 6, с использованием пакета php, базой данных Postgresql и пакетом для взаимодействия с базой nginx. Здесь, как и в дальнейшем, для примера будет использоваться структура проекта «Инфраструктурный сервис поддержки предпринимателей», разработанным на пятой версии фреймворка.

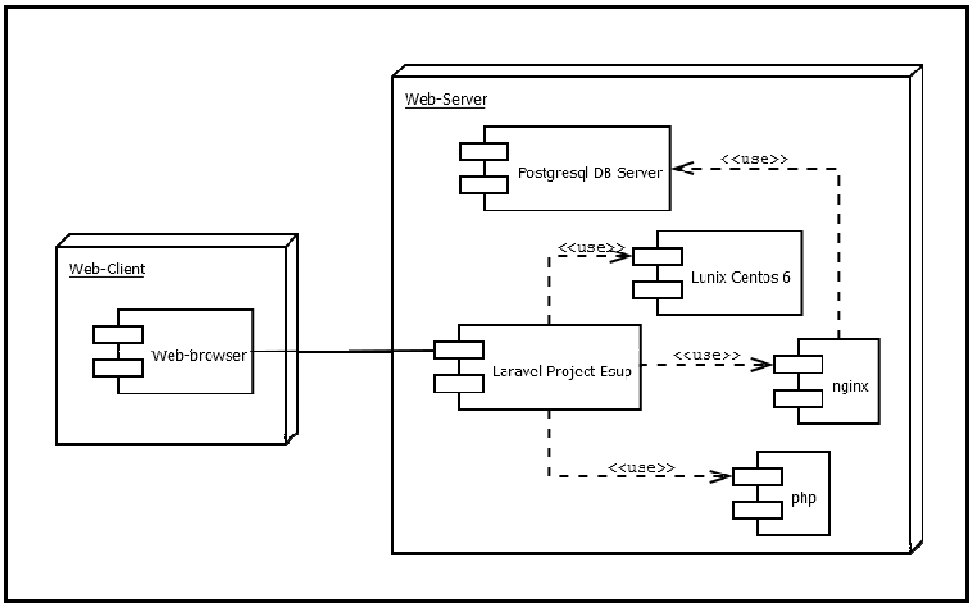

Рисунок 1. Диаграмма развертывания

Для взаимодействия клиента и сервера в Laravel 5.0 используются маршруты (запросы). Простейший маршрут состоит из url пути и функции замыкания. Существуют POST-запросы, предназначенные для принятия веб-сервером данных, и GET-запросы, используемые для запроса содержимого указанного ресурса. Большинство маршрутов определены в файле {папка проекта}/app/Http/routes.php. После получения сервером запрос фильтруется, и, в зависимости от результата, происходит вызов определенной функции контроллера. Рассмотрим этот процесс более подробно на примере диаграммы прецедента.

Прецедент в языке моделирования UML [1] обозначает сценарий использования. Диаграмма прецедента – схема последовательностей операций определенного прецедента. На рисунке 2 изображена диаграмма прецедента для просмотра веб-страницы проекта в продукте «Инфраструктурный сервис поддержки предпринимателей».

Для выполнения этого прецедента требуются три действующих лица – веб-браузер, являющийся клиентом, веб-проект laravel, расположенный на сервере и база данных для хранения информации.

Первое действие совершает клиент, а именно: при нажатии кнопки «Просмотреть проект» на веб-странице происходит отправка POST-запроса к серверу. Следует отметить, что веб-страницы в Laravel 5.0 также имеют определенную структуру. Для их отображения используются шаблоны содержащие HTML-код приложения и представляющие собой удобный способ разделения бизнес-логики и логики отображения информации. Шаблоны хранятся в папке ***{папка проекта}/resources/views***. Более подробно с шаблонами и другими использованными в данной статье возможностями фраймворка можно ознакомится в официальной документации [2], которая в данный момент практически полностью переведена на русский язык.

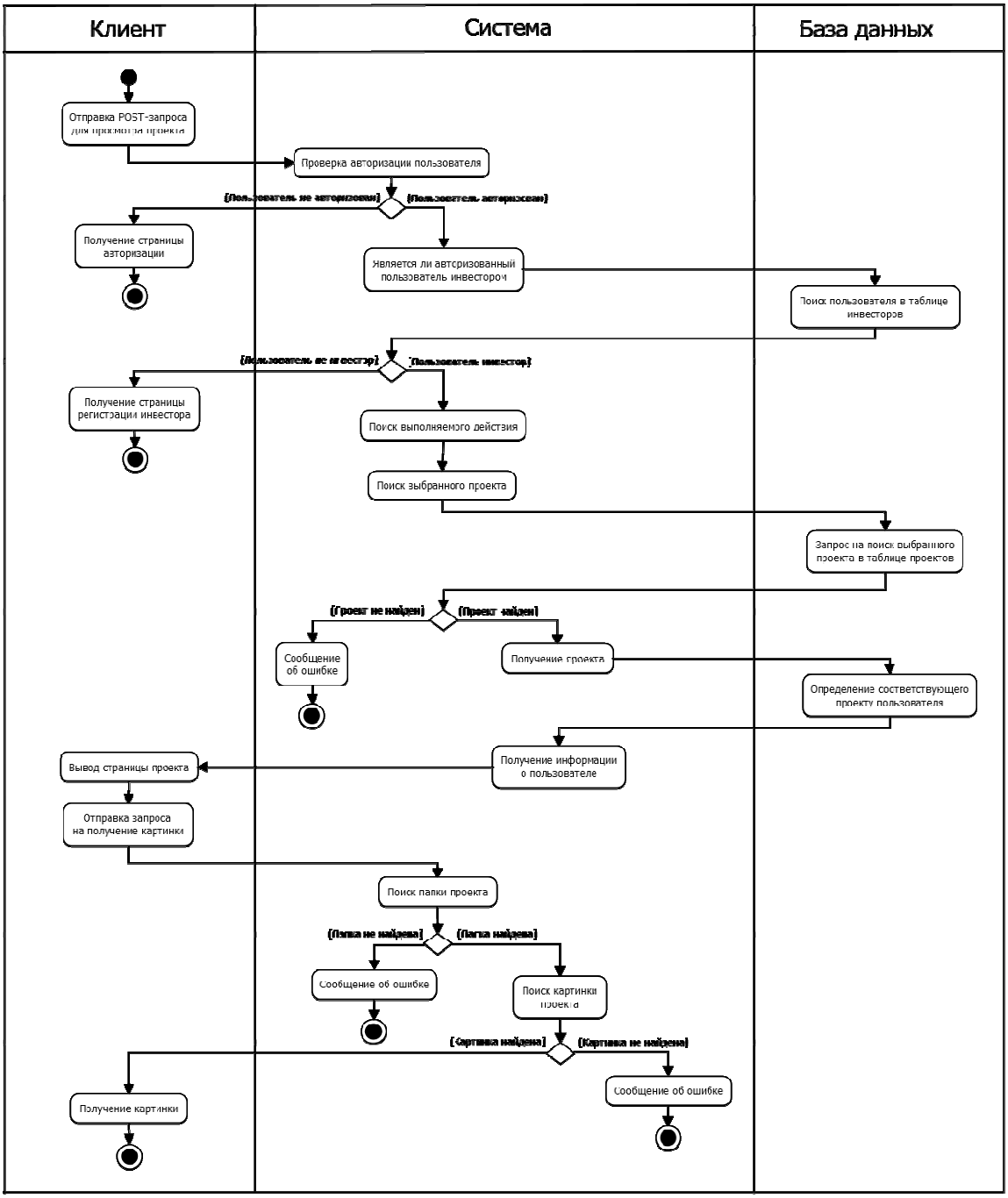

Рисунок 2. Прецедент «Просмотреть проект»

После получения запроса от клиента сервер проверяет его фильтром [3]. Некоторые маршруты, например показ страницы авторизации или регистрации, отображение главной страницы сайта, в большинстве случаев не фильтруют. Для того, чтобы запрос был обработан фильтром, в описание маршрута в файле routes.php добавляют код:

['middleware' => '{название фильтра}'].

В Laravel 5.0 существуют несколько дефолтных фильтров, расположенных в ***{папка проекта}/app/Http/Middleware***. Для создания собственного фильтра необходимо создать в этой папке класс с требуемым именем фильтра, зарегистрировать его в свойстве ***$middleware*** класса

app/Http/Kernel.php, а также добавить его в свойство ***routeMiddleware*** этого же класса, назначив ему некоторое имя, например, ***auth***, которое будет ключом массива.

Также для создания фильтра можно использовать интерфейс командной строки artisan, входящей в Laravel. Он предоставляет полезные команды для использования во время разработки веб-приложения.

Для создания фильтра пользуются командой:

php artisan make:middleware {название фильтра}.

В примере выше запрос проходит фильтр для проверки, авторизован ли пользователь. Для этого используется еще одна возможность Laravel 5.0 - система объектно-реляционного отображения ORM Eloquent. Достоинство этой системы заключается в многофункциональности и простоте настройки взаимодействия с базой данных. Для каждой таблицы создается соответствующий класс-модель, используемый для работы с этой таблицей. Все модели располагаются в ***{папка проекта}/app/***. Модель представляет собой класс, расширяющий класс ***Illuminate\Database\Eloquent\Model***. Пример текста модели:

class {название модели} extends Model {}.

Для создания модели можно также воспользоваться Artisan-командой:

php artisan make:model {название модели}.

Для проверки авторизации используется модель User, связанная с таблицей users в базе данных.

В результате обработки фильтром запроса сервер либо отправит пользователя на страницу авторизации, в случае, когда пользователь не авторизован, либо пропустит запрос дальше.

В случае, когда пользователь авторизован, вызывается фильтр, проверяющий, является ли активный пользователь инвестором. Для этого используется модель Investor, связанная с моделью User отношением один к одному [4], таким образом, одному инвестору соответствует один пользователь. Для этого также был использован инструмент Eloquent, который позволяет в теле модели указывать отношения между таблицами.

В результате обработки сервер либо отправит клиента на страницу регистрации инвестора, либо продолжит обрабатывать запрос.

Если пользователь авторизован и также является инвестором, значит, запрос принят и его необходимо обработать. В Laravel 5.0 для обработки запросов используются контроллеры [3]. Классы контроллеров обычно хранятся в папке ***{папка проекта}/app/Http/Controllers***, а связь с маршрутом указывается в файле routes.php:

Route::get('project/{id}', 'ProjectController@view');

Контроллеры создаются для обработки конкретных моделей, например для модели Project необходимо создать контроллер ProjectController, для связи с моделью в определении контроллера необходимо указать зависимость:

use App/{название модели}.

В рассматриваемом примере следующим действием будет выполнение связанной с запросом на просмотр функции в контроллере проектов.

Этой функцией производится поиск модели проекта с требуемым идентификатором в базе данных. В случае, когда проект с указанным идентификатором не найден, сервер отправляет клиенту сообщение об ошибке, иначе производится поиск информации о пользователе, разместившем проект, и сервером генерируется страница с требуемым проектом.

В HTML-коде страницы присутствует вызов запроса на отображение картинки, связанной с проектом.

Для этого используется современный механизм работы с файловой системой Flysystem. Flysystem позволяет легко переключиться с хранения файлов на сервере на хранение файлов в облачных хранилищах. Настройки Flysystem находятся в файле ***{папка проекта}/config/filesystems.php***. Внутри него можно настроить несколько дисков, каждый из которых представляет свой тип хранения - локальная файловая система или облачные хранилища. Для взаимодействия с файловой системой использовался фасад Storage, позволяющий хранить все файлы проекта в локальном хранилище сервера, доступ к которому можно ограничить фильтрами запросов.

В рассматриваемом проекте для каждого проекта создается папка, в которой хранится картинка проекта и некоторые документы. Поэтому после получения запроса от клиента сервер сначала проверяет наличие папки проекта, а уже после наличие картинки. В случае отсутствия пользователю выводится сообщение об ошибке.

Помимо указанных в примерах в Laravel 5.0 присутствует еще много других возможностей, или сервисов, доступных из коробки. Это и механизм аутентификации, и сервис для отправки писем, и система кеширования, обработчик ошибок, валидация данных, а также много других.

В декабре 2013 года на сайте sitepoint.com был проведен опрос о самых популярных php-фреймворках, в котором Laravel признали самым многообещающим проектом на 2014 год. В 2015 году в рамках опросах на том же сайте занял первое место в номинациях: фреймворк корпоративного уровня и фреймворк для личных проектов.

Список литературы

1. OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2 [Электронный ресурс]. – http://doc.omg.org/formal/2007-11-02.pdf. – (дата обращения: 03.12.2015).

2. Документация Laravel 5.0 [Электронный ресурс]. – http://laravel.su/docs/5.0. – (дата обращения: 03.12.2015).

3. Апальков В. В. Основы моделирования цифровой обработки сигналов в среде MATHLAB: учебное пособие / В. В. Апальков, Р. А. Томакова. Курск, 2015. – 137 с.

4. Томакова Р. А. Методы и алгоритмы теории принятия решений: учебное пособие / Р. А. Томакова, В. В. Апальков. Курск, 2015. – 164 с.