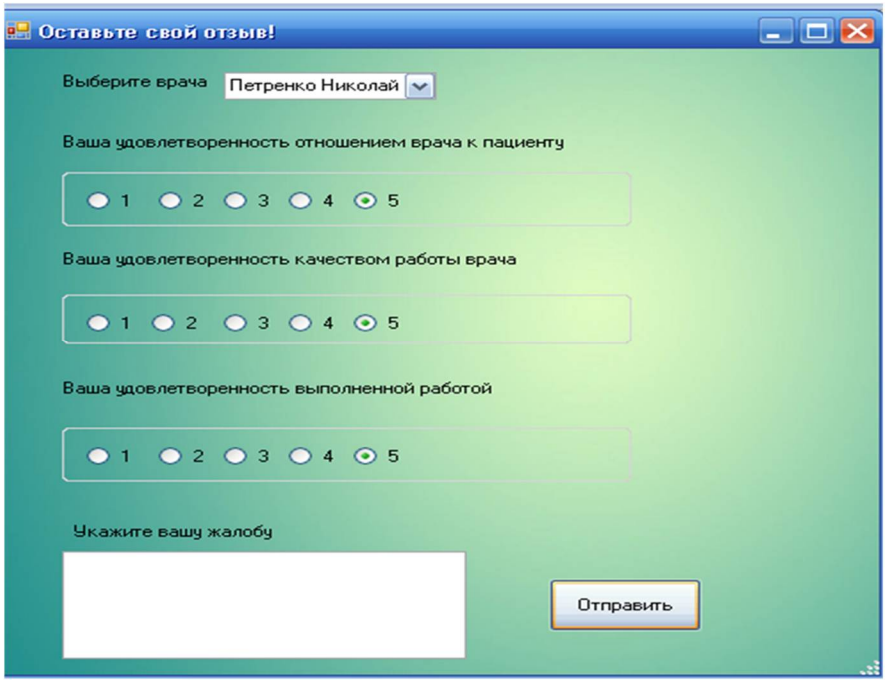

Рис. 5. Добавление отзыва

В заключении стоит отметить, что благодаря разработанному программному продукту повышается эффективность бизнес-процессов клиники. Это приводит к росту выручки от предоставляемых услуг и увеличению прибыли. В результате конкурентоспособность клиники увеличивается. Областью применения данного программного продукта так же могут стать как частные, так и государственные клиники.

Список литературы

1. URL:http://resgroup.ru/content/fransh/automation.php

2. Карпов О. Э. Автоматизация бизнес-процессов лечебного учреждения на основе медицинской информационной системы (Национальный медико-хирургический Центр имени Н.И. Пирогова Министерства здравоохранения Российской Федерации. Москва, Россия), 2016.

3. Software Engineering компьютерных систем. Парадигмы, технологии и CASE-средства программирования / Е. М. Лаврищева. К. :Наук. думка. 2013

УДК 004.432

ПОСТРОЕНИЕ REST API НА ОСНОВЕ ВЕБ-ФРЕЙМВОРКА LARAVEL

Т. Л. Тен*, Н. А. Шинекенев**, Г. Д. Когай**

**Карагандинский экономический университет,*

***Карагандинский государственный технический университет*

(Республика Казахстан)

Веб-сервисы являются стандартом для обмена данными и интеграции между разными системами. Restful API стал основной парадигмой развития веб-сервисов после SOAP, а в

эффективном создании Restful API все еще остаются исследовательские задачи. В данной статье представлена модель разработки Restful API, основанная на языке PHP и структуре Laravel. Рассматриваются основные технические проблемы, которые необходимо решить при построении Restful API, и приведены детали реализации на основе Laravel.
Ключевые слова: *REST, API, Laravel, PHP, веб-сервис, данные, микросервис.*

Web services are a standard for data exchange and integration between different systems. Restful API has become the main paradigm for the development of web services after SOAP, and research tasks are still in the effective creation of the Restful API. This article presents the Restful API development model based on the PHP language and the Laravel structure. The article considers the main technical problems that need to be solved when building the Restful API. Also article gives the implementation details based on Laravel.
Keywords: *REST, API, Laravel, PHP, web service, data, microservice.*

В эпоху индустрии программного обеспечения, веб-приложения имеют все более растущую популярность, которая подтверждается тем, что многие из них могут обслуживать миллионы пользователей в день. Архитектура монолитного приложения не подходит для современных масштабируемых веб-приложений, особенно в распределенных вычислительных средах, и несовместима с моделью командной работы над приложением. В обозримом будущем, из-за быстрого развития веб-приложений и Интернета вещей, архитектура приложений должна станет более плоской, а распределенные вычисления станут основной моделью разработки программного обеспечения. В этом контексте нам нужна простая и надежная разнородная модель разработки микросервиса. При создании приложения на основе микросервиса, разработчик имеет возможность больше сосредоточиться на бизнес-процессах.

REST - архитектурный стиль взаимодействия компонентов распределенного приложения в сети. Использование REST обусловлено простотой, масштабируемостью, эффективностью и безопасностью архитектуры разрабатываемого приложения. REST - это протокол RPC, построенный на основе протокола HTTP. Простота и удобство использования делает его непревзойденной альтернативой SOAP, популярному RPC-решению прошлых лет. Структурированное веб-приложение на основе Rest легко создается, и многие разработчики успешно реализуют проекты создавая API на комбинациях технологии Ajax и Restful [1, с. 17].

Репрезентативная передача состояния (REST) или веб-службы RESTful - это способ обеспечения взаимодействия между компьютерными системами в Интернете. REST-совместимые веб-службы позволяют системам, которые запрашивают данные, получать доступ к текстовым представлениям веб-ресурсов и управлять ими, используя единый и предопределенный набор операций.

Термин репрезентативная передача состояния (REST) был введен и определен в 2000 году Роем Филдингом в его докторской диссертации. Филдинг использовал REST для разработки HTTP 1.1 и Uniform Resource Identifiers (URI). Ресурс - это определенная информация, к которой можно получить доступ, например, это может быть объект приложения, запись базы данных или

алгоритм. Каждый ресурс идентифицируется уникальным URI (Universal Resource Identifier). REST представляет URI в формате «/user/name», а операции HTTP-методами - GET, PUT, POST, DELETE, HEADER и OPTIONS, передавая информацию обратно к компьютеру-клиенту. Важной характеристикой REST является то, что серверная сторона сохраняет изначальное состояние между несколькими взаимодействиями, каждый сервер в кластерах может обслуживать клиента по каждому запросу [2, с.16].

Современное приложение для обслуживания веб-сервисов, основанное на Laravel, должно состоять из следующих компонентов: механизм маршрутизации, промежуточные обработчики, паттерн MVC, ORM и компонент аутентификации. Когда клиент отправляет запрос в Laravel Restful API через HTTP, веб-сервер сначала получает запрос и передает его в PHP, тогда фактическое выполнение начинается с Larvel для инициализации подпрограмм. Процедуры инициализации Laravel завершают некоторую настройку и затем запрос передается для фильтрации в промежуточные обработчики. После фильтрации, в дело вступает компонент маршрутизации, отвечающий за диспетчеризацию запроса в соответствии с таблицей конфигурации маршрута. В конце, контроллер MVC берет на себя отправленный запрос. Контроллер отвечает за выполнение логики приложения и запрашивает модель для извлечения данных и состояния, а компонент представления берет на себя последнюю часть работы для отображения страницы. Однако, когда создается Restful API, рендеринг страниц не требуется, поэтому задействованы только контроллер и модель. В конце процесса будет возвращаться только данные, закодированные с помощью JSON. Так как модель сильно зависит от ORM для обработки данных, Laravel имеет встроенный инструмент ORM для управления реляционной базой данных - Elequent.

ORM (Object Relational Mapping) - это метод программирования для преобразования данных между базой данных и объектами в памяти при использовании объектно-ориентированных языков программирования. Это фактически создает «базу данных виртуальных объектов», которую можно манипулировать с помощью языка программирования. Модификация будет автоматически синхронизирована с реальной базой данных. Eloquent ORM, входящий в состав Laravel, обеспечивает лаконичную и простую реализацию ActiveRecord для работы с базой данных. Каждая таблица базы данных имеет соответствующую «Модель», которая используется для взаимодействия с этой таблицей. Модели позволяют запрашивать данные в таблицах, а также вставлять новые записи в таблицу. В пакете Eloquent ORM определен базовый класс «Model», который содержит множество методов и утилит для работы с базой данных. Все модели, специфичные для приложения, должны наследоваться от этого базового класса [3,с. 67].

Все маршрутизаторы системы находятся в каталоге routes, а определения маршрутов автоматически загружаются и анализируются во время инициализации. Простая схема маршрутизаций может состоять только из URI и замыкания, которое выполняется всякий раз, когда запрашивается

URI. Более общим способом определения правил маршрутизации является массив конфигурации в «routes/api.php», где каждое правило добавляется, когда элемент конфигурации находится в массиве.

Одна строка вызова функции «Маршрут :: ресурс» определяет несколько действий маршрутизации, как показано в следующей таблице. Когда эти правила маршрутизации вступят в силу, запрос на URL-адрес «http://www.вашдомен.com/path» будет отправлен на контроллер с именем «PathController», и функция «index()» будет вызвана для обработки запроса.

Таблица 1

Маршруты URL в приложении на основе Laravel

Метод	Маршрут URL	Действие	Название маршрутизатора
GET	/path	index()	path.index
GET	/path/create	create()	path.create
POST	/path	store(Request $request)	path.store
GET	/path/{args}	show($args)	path.show
GET	/path/{args}/edit	edit($args)	path.edit
PUT/PATCH	/path/{args}	update(Request $request,$args)	path.update
DELETE	/path/{args}	destroy($args)	path.destroy

Архитектура микросервиса уже является фактическим отраслевым стандартом для веб-сервиса, а веб-сервисы Restful стали предпочтительной технологической моделью для приложений для микросервисов благодаря своей легкости, масштабируемости и совместимости с протоколом HTTP.

Список литературы

1. Отвелл Тайлер. Laravel. URL: https://laravel.com/
2. Каллум Хопкинс. Шаблон MVC и PHP, часть 1. URL: https://www.sitepoint.com/the-mvc-pattern-and-php-1/. 2013.
3. Сергей Рогачев. Обобщенный Model-View-Controller // RSDN.ORG. 2007.
4. Джо Стамп. Понимание MVC в PHP. // O'Reilly Media, Inc. 2015.
5. Хардик Пател. Преимущества PHP MVC фреймворков. // URL: https://www.linkedin.com/pulse/advantages-php-mvc-framework-why-laravel-become-most-2017-hardik.