Restful API Architecture Based on Laravel Framework

Restful API Architecture Based on Laravel Framework

Xianjun Chen[1,a], Zhoupeng Ji[1], Yu Fan[1,b,*] and Yongsong Zhan[2]

[1]Haikou College of Economics, Haikou, China
[2]Guilin University of Electronic Technology, Guilin, China
Email: [a]hingini@126.com, [b]149471869@qq.com

Abstract. Web service has been an industry standard tech for message communication and integration between heterogeneous systems. RESTFUL API has become mainstream web service development paradigm after SOAP, how to effectively construct RESTFUL API remains a research hotspots. This paper presents a development model of RESTFUL API construction based on PHP language and LARAVEL framework. The key technical problems that need to be solved during the construction of RESTFUL API are discussed, and implementation details based on LARAVEL are given.

1. Introduction

In the age of a rapid development software industry, the internet application has got unstoppable growing popularity among which many of them can serve millions of users per day and reach a daily PV of billion level. Monolithic application architecture is not suitable for today's large-scale Internet applications, especially in distributed and heterogeneous computing environments, nor does it satisfied with multi-team collaborative development model. In the foreseeable future, due to the failure of Morgan's theory, the rapid development of Internet applications and Internet of things, the architecture of applications should get more flatten, distributed computing will become the mainstream software development model. In such context, we need a simple and reliable heterogeneous micro-service development model. When we build an application on some micro service infrastructures, the developer can focus on the business process and leave out to the framework all the complex about the inter-process communication.

2. API Technology

REST is a technical specification about heterogeneous inter-communication for Web applications. Adoption of REST can lead to a simple, scalable, effective, safe and reliable architecture. REST is a lightweight RPC protocol built upon HTTP protocol, it's simplicity and web friendliness makes it an unparalleled alternative to SOAP, a popular RPC solution for years. A well-structured Web application can be easily built when using Rest, Many developers successfully created simple and robust API base on Ajax and Restful web service.

2.1 Restful API

Representational state transfer(REST) or RESTful web services is a way of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations[2].

The term representational state transfer was introduced and defined in 2000 by Roy Fielding in his doctoral dissertation[3][4]. Fielding used REST to design HTTP 1.1 and Uniform Resource Identifiers (URI)[3][5][6]. A resource is a kind of information that can be accessed, such as an application

object, a database record, an algorithm, and so on. Each resource is identified by a unique URI (Universal Resource Identifier), REST represent URI in the form of "/user/name", and operations on HTTP methods GET, PUT, POST, DELETE, HEADER and OPTIONS, resulting in the next resource being transferred back to the caller. An important characteristic of REST is that the server side keeps stateless between multiple interactions, every server in the clusters can serve the client on every request.

2.2 LNMP Stack

LNMP is an archetypal model of web development stacks, named as an acronym of the names of its original four open-source components: the Linux operating system, the Nginx HTTP Server, the MySQL relational database management system (RDBMS), and the PHP programming language. As a solution stack, LAMP is suitable for building dynamic web sites and web applications.

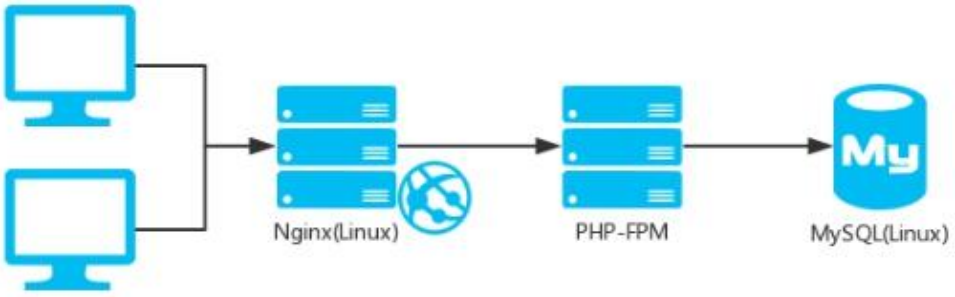

Figure 1 LNMP Stack

2.3 Laravel Framework

Laravel[1] is a free, open-source PHP web framework, created by Taylor Otwell and intended for the development of web applications following the model–view–controller(MVC) architectural pattern. Some of the features of Laravel are a modular packaging system with a dedicated dependency manager.

The laravel framework is easy to understand and powerful, the framework itself provides authentication, routing, session manager, caching, IoC container and tons of most commonly used component, also amazing database migration tools and integrated unit testing support, all these tools give developers the ability to build complex applications.

3. System Architecture design

A sophisticated Laravel based restful web service application should comprise of some key component, such as routing engine, middleware, MVC framework, ORM and authentication component.

When a client sends a request to Laravel restful API through HTTP, the web server first receives the request and passing through to PHP Engine, then the actual execution start from Laravel initialize routines. The Laravel initialize routines finish some configuration, the request is then passed To middleware engine for filtering, after filtering there comes the core routing component responsible for request dispatching according to the route configuration table, finally the MVC controller take over the dispatched request. In general, the controller is responsible for application logic execution and inquire the model for data fetch and persistent, the view component takes over the last piece of work for rendering the page. But when we build Restful API, page rendering is useless, so only controller and model are involved, at the end of the process is only Data encoded with JSON will be returned. Laravel provides a powerful ORM framework for relational database manipulate, the model depends heavily rely on ORM for data processing. The whole process are shown as figure 2.

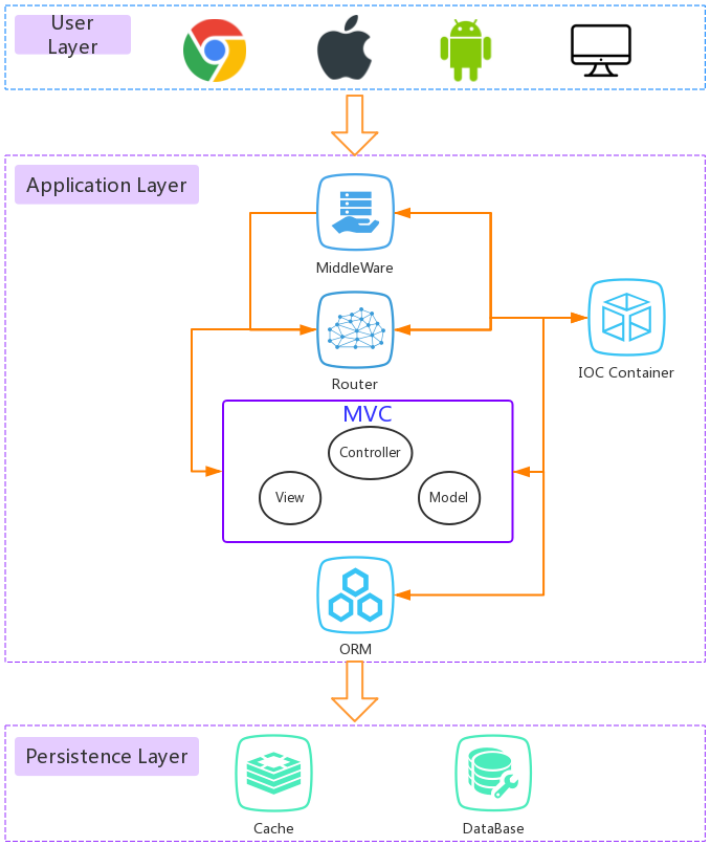

Figure 2 System Architecture

3.1. Middleware

The main purpose of HTTP middleware is to provide a convenient mechanism for filtering HTTP requests before the actual logic process, middlewares chain together to do filter process one by one. These middleware handle reading and writing the HTTP session, determining if the application is in maintenance mode, verifying the CSRF token, and more.

Two kinds of middleware exist in Laravel, one is for HTTP filtering, the other is for routing filtering. The request can hit the controller only after it can pass through all the HTTP middlewares and then routing middleware.

The core component drive Laravel is the class "\ Illuminate \ Foundation \ Http \ Kernel", it acts as a kernel serve in the central location that all requests flow through, the HTTP kernel also defined a list of HTTP middleware that all requests must pass Through all being handled by the application. All middlewares depend on one base class "Illuminate \ Pipeline \ Pipeline", middlewares can form a chainable call stack, benefit from the chainable closure implementation define the base class.

3.2. Routing

All routers[7] of the system are located in the directory named routes, routing definitions are automatically loaded and parsed during initialization. The simplest routing can only accept a URI and a closure, which is executed whenever the URI is requested, but it's a hard coding style. A more common way of define rules of routing is the configuration array in "routes/api.php", each rule is adding when a configuration item is found in the array, the code is as below:

Route::resource('path','PathController');

A single line of "Route::resource" function call define multiple routing actions as shown in the following table. When these routing rules take effect, a request for URL "http://www.yourdomain.com/path" will be dispatched to a controller named "PathController" and the "Index()" function will be called to handle the request.

Table1 URL Pattern

Method	URL Pattern	Action	Router Name
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

3.3. ORM

ORM (Object Relational Mapping) is a programming technique for converting data between database and objects in memory when using object-oriented programming languages. This creates, actually, a "virtual object database" that can be manipulated from within the programming language, and all The modification will be automatically synchronized to the real database in the hard disk.

The Eloquent ORM included with Laravel provides a beautiful, simple ActiveRecord implementation for working with your database. Each database table has a corresponding "Model" which is used to interact with that table. Models allow you to query for data in your tables, as well as insert new records into the table. In Eloquent ORM package, a base class "Model" is defined which contains many database operation methods and utility tools, all application specificated model should inherit from this base class.

The Eloquent model is usually placed in the app directory and can be automatically loaded via composer.json. Usually "Illuminate\Database\Eloquent\Model" come as base， and the subclass definition as：

```
class User extends Model {}
```

Eloquent will map the above "User" model the underline database correspondence of "users" table, so any operations on"User" model will directly affect "users" table.This default behavior can be overridden if a specificated data table name is supplied, as shown in the following code:

```
class User extends Model
{
protected $table = 'user_table';
}
```

4. JWT and Authentication

JSON Web Token (JWT) is a JSON-based open standard for creating access tokens that assert some number of claims.Tokens are signed by the server's key for legitimate verification in both client and server side. The tokens are designed to be compact, URL-safe, and typically be used to pass the identity of authenticated users between an identity provider and a service provider.

4.1. JWT in Laravel

An extension package named "tymon/jwt-auth" developed by the community can be integrated into to Laravel framework which will provide JWT functionality, we can install it from composer on the command below:

```
composer require tymon/jwt-auth
```

After installaiotn, "config/app.php" should be configured for service provider and aliase：

```
'providers' =>
 [
        Tymon\JWTAuth\Providers
        \JWTAuthServiceProvider::class,
 ],
'aliases' =>
 [
```

```
        'JWTAuth'=> Tymon\JWTAuth\Facades
        \JWTAuth::class
 ],
```

And，guards should be set in "config/auth.php"：

```
'api' =>
[
        'driver' => 'token',
        'provider' => 'users',
],
```

Finally，all these should be assembled together in a route group setting to get applied：

```
Route::group(
        ['middleware'=>['api'],'
        prefix'=>'api'],
        function(){
        Route::post(api1, 'ApiController@api1);
        Route::post(api2, 'ApiController@api2);
         }
);
```

5. Cross-domain Solution

A cross-domain request original from same-origin policy, a web application security model of webkit based browser which permits scripts contained in a first web page to access data in a second web page, but only if both web pages have the same origin. The so-called same-origin refers to the domain name, protocol and port are all the same.

Cross-origin resource sharing (CORS) is a mechanism that allows restricted resources on a web page to be requested from another domain outside the domain from which the first resource was served，both browser and server should be supported for CORS to take effect. In server side it is common to set the Access-Control-Allow-Origin attribute in the HTTP header to specify which sites can access。

Since Laravel provides a good middleware extension mechanism, we can implement cross-domain filtering through middleware, the code example is as follows:

```
<?php namespace App\Http\Middleware;
use Closure;
use Response;
class EnableCrossRequestMiddleware
{
    public function handle($request, Closure $next)
    {
        $response = $next($request);
        $response->header('Access-Control-Allow-Origin', '*');
        $response->header('Access-Control-Allow-Headers', 'Origin, Content-Type, Cookie, Accept');
        $response->header('Access-Control-Allow-Methods', 'GET, POST, PATCH, PUT, OPTIONS');
    }
}
```

6. Conclusion

Micro service architecture is already a de facto industry standard for web service, and Restful Web Services has become the preferred technology model for micro services application due to its lightweight, scalability, and HTTP protocol compatibility, This paper discusses the conception of how to use Laravel framework to build the basic infrastructures of the Restful service and provides key implementation details for our solution.

7. Acknowledgments
This work has been supported by the grant of Hainan Natural Science Foundation(NO:20156239), the grant of Science Research Foundation of Haikou College of Economics(NO: hjkz16-02) . Corresponding author: Fan Yu(149471869@qq.com)

8. Reference

[1] https://laravel.com/
[2] "Web Services Architecture". World Wide Web Consortium. 11 February 2004. 3.1.3 Relationship to the World Wide Web and REST Architectures. Retrieved 29 September 2016.
[3] Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST)".
[4] "Fielding discussing the definition of the REST term". groups.yahoo.com. Retrieved 2017-08-08.
[5] RFC 1945
[6] RFC 2616
[7] Zhou Qiao Jun. The Design and Implementation of RESTful Web Services Open Platform[D].Zhejiang University,2016.
[8] Tang Ming Wei. Research and Implementation of Library Management System Based on REST Architecture[J]. New Technology of Library and Information Service,2010,(09):84-89.
[9] Xue Min. Design and Implementation of P2P Network Loan System Based on Laravel Framework[D].JiLin University,2016.
[10] Chen Yan Hui. Shipping Information Interaction Model and Its Application Based on Saa S and Laravel Architecture[D].Zhejiang University of Technology,2016.
[11] Qin Fen. Research And Design Of Mobile Environment For Restful Webservice [D].Beijing industry university,2015.