

**Composer. Lumen by Laravel.
RESTful API example.**

Composer

Composer – менеджер зависимостей для PHP.

Разработчик описывает от каких библиотек зависит проект. Composer – устанавливает нужные библиотеки.

Аналог для js – npm.

Composer на Openserver

- Скачиваем [Composer-Setup.exe](#)
- На этапе установки указываем путь
\\PATH\\OpenServer\\modules\\php\\PHP-5.6\\php.exe
- В настройках Openserver, вкладка модули, выставляются значения:
- HTTP: Apache-2.4
- PHP: PHP-5.6

Composer на Openserver

- Openserver. Дополнительно -
Конфигурация – PHP. Нужно
удостовериться, что строка
`extension=php_openssl.dll`
раскомментирована.

Composer на Openserver

Openserver. Дополнительно – консоль.

Перейти к папке с модулем php 5.6:

```
cd modules/php/PHP-5.6
```

Затем выполнить:

```
php -r "readfile('https://getcomposer.org/installer');" | php
```

В случае успеха:

Composer succesfully installed to ...

Composer на Openserver

Проверить установку composer:
`php composer.phar -V`

Результат:

`Composer version <версия> < дата
обновления >`

Composer на Openserver

Можно выполнить команду

```
echo @php "%~dp0composer.phar" %*>composer.bat
```

Тогда доступ к composer будет короче:

```
composer -V
```

Lumen от Laravel. Микрофреймворк.

Требования к серверу:

- PHP \geq 5.5.9
- OpenSSL PHP Extension
- PDO PHP Extension
- Mbstring PHP Extension

Lumen от Laravel. Установка

- Openserver. Дополнительно – консоль.
- `cd domains/test`
- `php composer create-project laravel/lumen`
- Будет создана папка lumen в каталоге с сайтом.

Lumen от Laravel. Структура

▼ simple-rest-api

▼ lumen

- ▶ app
- ▶ bootstrap
- ▶ database
- ▶ public
- ▶ resources
- ▶ storage
- ▶ tests
- ▶ vendor

.env

.gitignore

artisan

composer.json

composer.lock

phpunit.xml

readme.md

Lumen от Laravel. Настройки доступа к БД.

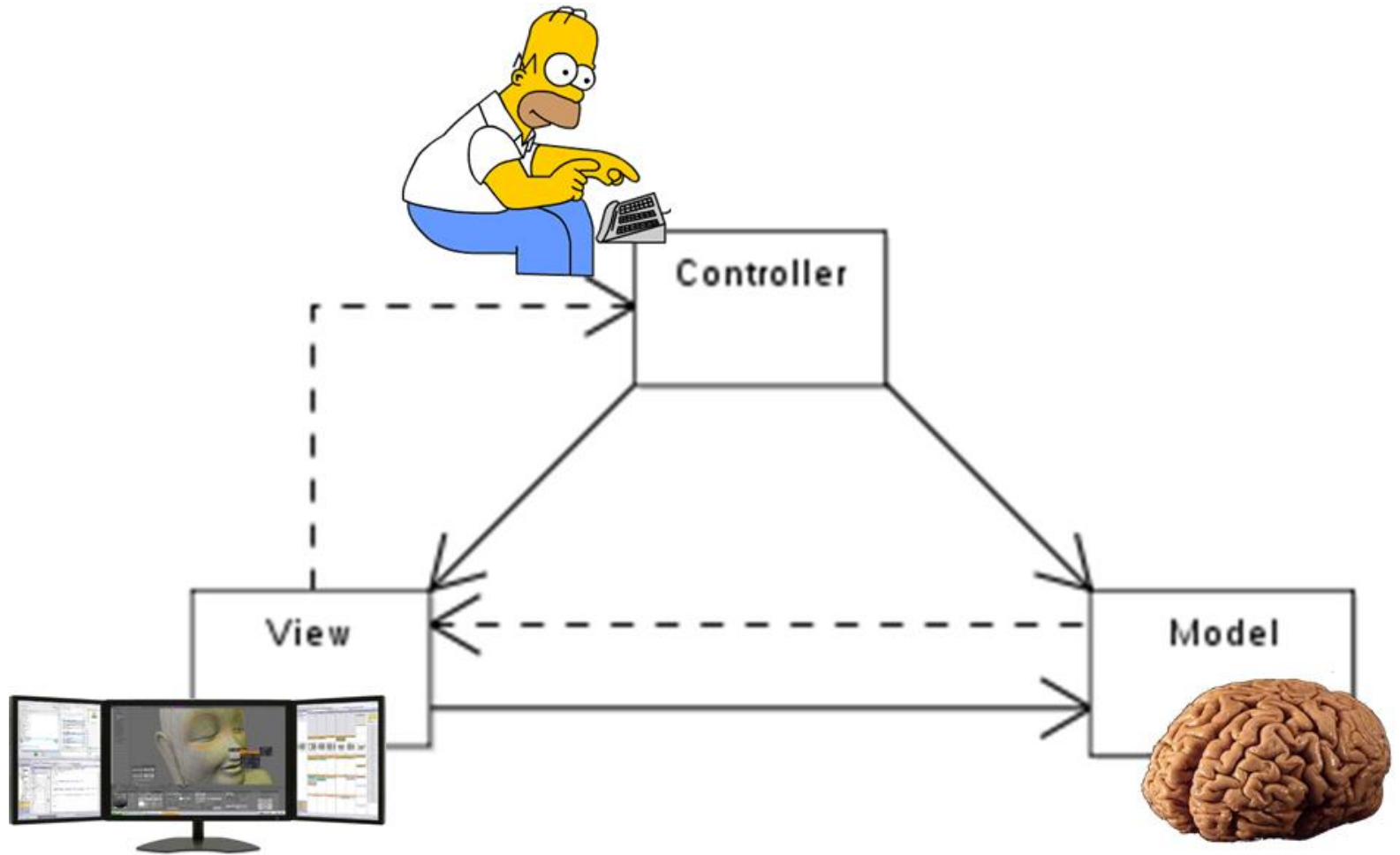
- Переименовать **.env.example** в **.env**
- Прописать в env настройки доступа к бд (базу предварительно создать в phpMyAdmin)

```
APP_ENV=local
APP_DEBUG=true
APP_KEY=srapp123

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=rest_test
DB_USERNAME=root
DB_PASSWORD=''

CACHE_DRIVER=memcached
QUEUE_DRIVER=database
|
```

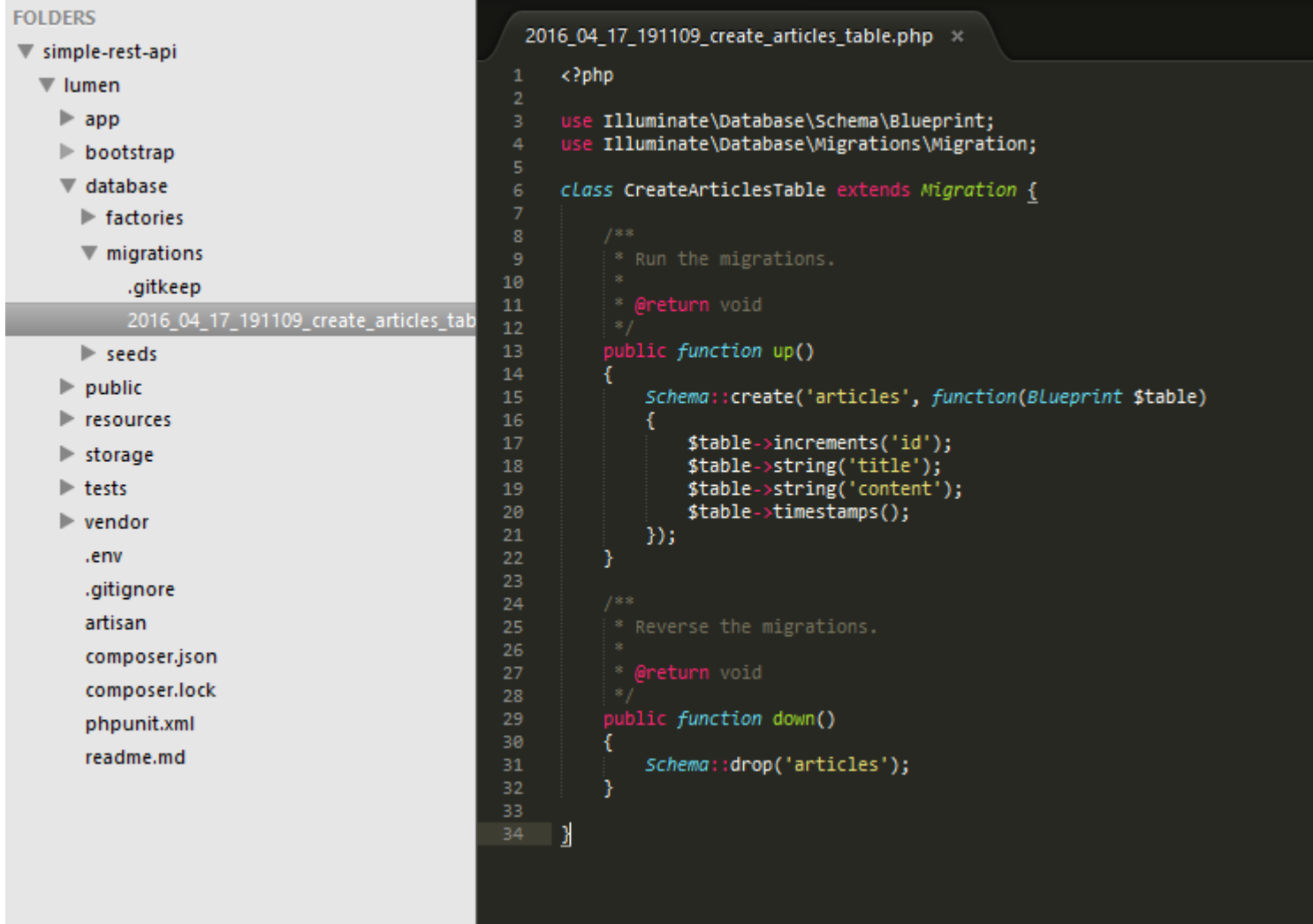
MVC



Lumen от Laravel. Создание модели

- Дополнительно – консоль
- `cd domain/test/lumen`
- `php artisan make:migration
create_articles_table --create=articles`

Lumen от Laravel. Создание модели



FOLDERS

- ▼ simple-rest-api
 - ▼ lumen
 - ▶ app
 - ▶ bootstrap
 - ▼ database
 - ▶ factories
 - ▼ migrations
 - .gitkeep
- 2016_04_17_191109_create_articles_tab
 - ▶ seeds
- ▶ public
- ▶ resources
- ▶ storage
- ▶ tests
- ▶ vendor
- .env
- .gitignore
- artisan
- composer.json
- composer.lock
- phpunit.xml
- readme.md

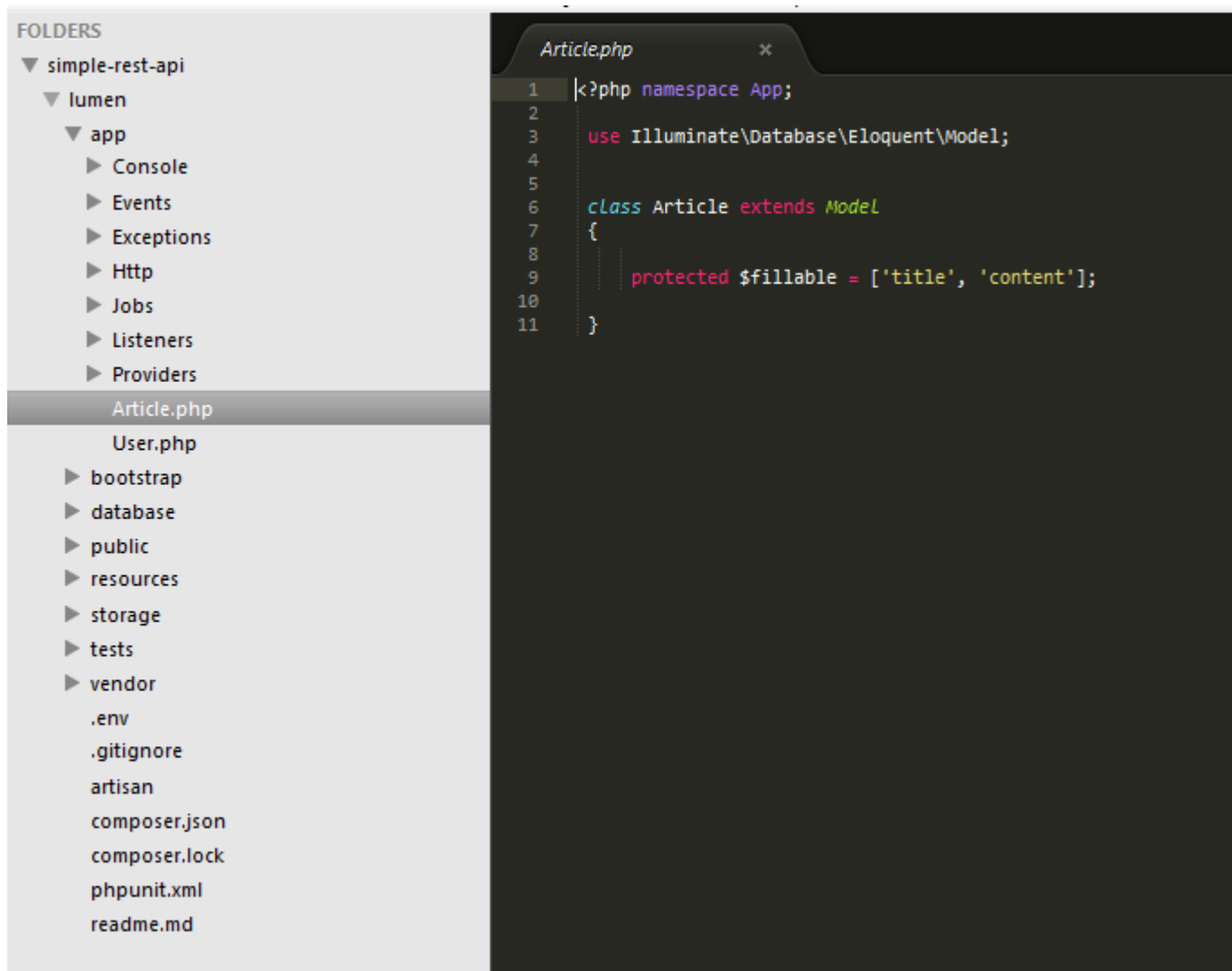
2016_04_17_191109_create_articles_table.php ×

```
1  <?php
2
3  use Illuminate\Database\Schema\Blueprint;
4  use Illuminate\Database\Migrations\Migration;
5
6  class CreateArticlesTable extends Migration {
7
8      /**
9       * Run the migrations.
10      *
11      * @return void
12      */
13      public function up()
14      {
15          Schema::create('articles', function(Blueprint $table)
16          {
17              $table->increments('id');
18              $table->string('title');
19              $table->string('content');
20              $table->timestamps();
21          });
22      }
23
24      /**
25       * Reverse the migrations.
26       *
27       * @return void
28       */
29      public function down()
30      {
31          Schema::drop('articles');
32      }
33
34  }
```

Lumen от Laravel. Создание модели

- `php artisan migrate`
- После выполнения этой команды в базе данных будет создана таблица `articles`, структура которой задается в функции `up()`.
- В каталоге `app` создаем файл модели `Article.php`

Lumen от Laravel. Создание модели

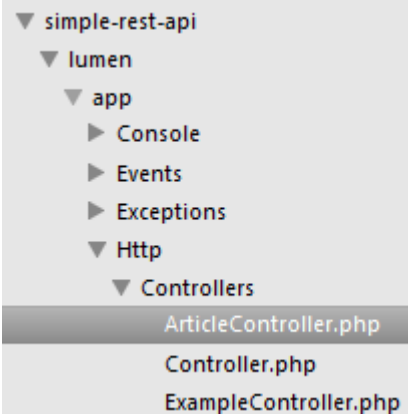


The screenshot displays an IDE interface. On the left, a 'FOLDERS' sidebar shows the project structure: 'simple-rest-api' > 'lumen' > 'app'. The 'app' folder is expanded, showing subfolders like 'Console', 'Events', 'Exceptions', 'Http', 'Jobs', 'Listeners', and 'Providers'. Below these, 'Article.php' is selected. The main editor area shows the code for 'Article.php'.

```
Article.php
1 |<?php namespace App;
2
3 | use Illuminate\Database\Eloquent\Model;
4
5
6 | class Article extends Model
7 | {
8 |
9 |     protected $fillable = ['title', 'content'];
10
11 | }
```


Lumen от Laravel. Создание контроллера

- В каталоге
lumen/app/Http/Controlle
rs/ создаем файл
ArticleController.php



```
▼ simple-rest-api
  ▼ lumen
    ▼ app
      ► Console
      ► Events
      ► Exceptions
      ▼ Http
        ▼ Controllers
          ArticleController.php
          Controller.php
          ExampleController.php
```

Lumen от Laravel. Создание контроллера

```
<?php

namespace App\Http\Controllers;

use App\Article;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;

class ArticleController extends Controller{

    public function index(){

        $articles = Article::all();

        return response()->json($articles);

    }

    public function getArticle($id){

        $article = Article::find($id);

        return response()->json($article);

    }

}
```

Lumen от Laravel. Создание контроллера

```
public function saveArticle(Request $request){  
    $article = Article::create($request->all());  
    return response()->json($article);  
}  
  
public function deleteArticle($id){  
    $article = Article::find($id);  
    $article->delete();  
    return response()->json('success');  
}  
  
public function updateArticle(Request $request,$id){  
    $article = Article::find($id);  
    $article->title = $request->input('title');  
    $article->content = $request->input('content');  
    $article->save();  
    return response()->json($article);  
}
```

Lumen от Laravel. Включаем ORM

- В файле `/lumen/bootstrap/app/` раскомментируем строки

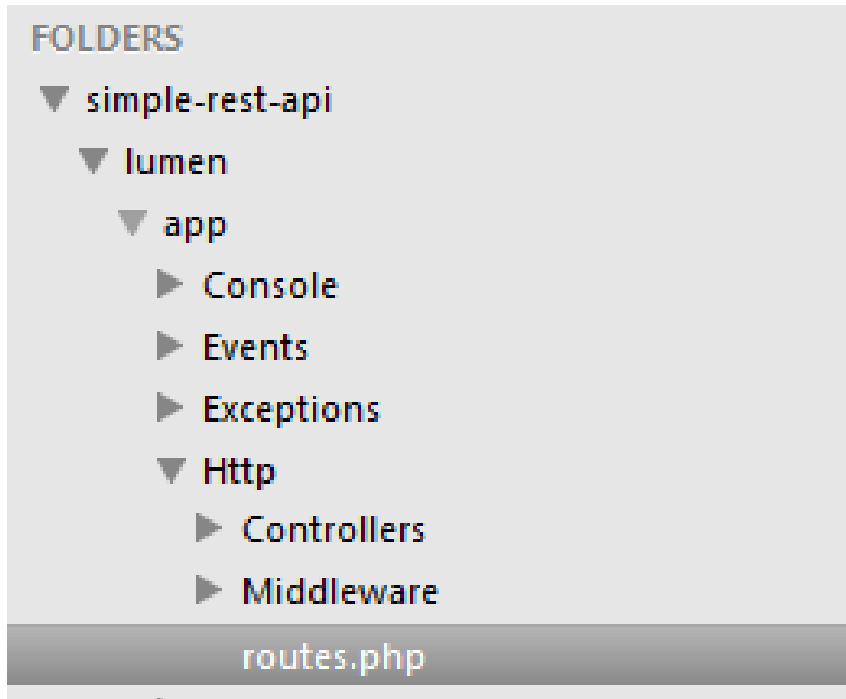
```
$app->withFacades();
```

```
$app->withEloquent();
```

Lumen от Laravel.

- В файле `/lumen/public/index.php` заменим `$app-run();` на `$app->run($app->make('request'));`

Lumen OT Laravel. Routing.



Lumen or Laravel. Routing.

```
<?php

/*
|-----
| Application Routes
|-----
|
| Here is where you can register all of the routes for an application.
| It is a breeze. Simply tell Lumen the URIs it should respond to
| and give it the Closure to call when that URI is requested.
|
*/

$app->get('/', function () use ($app) {
    ... return $app->version();
});

$app->get('api/article', 'ArticleController@index');

$app->get('api/article/{id}', 'ArticleController@getArticle');

$app->post('api/article', 'ArticleController@saveArticle');

$app->put('api/article/{id}', 'ArticleController@updateArticle');

$app->delete('api/article/{id}', 'ArticleController@deleteArticle');
```

Lumen от Laravel. Проверка

- Расширение на chrome Postman



Lumen от Laravel. Проверка

The screenshot shows a REST client interface with a list of requests at the top. The first request is a GET request to `http://simple-rest-api/lumen/public/api/article/1`. Below the list, the details of this request are shown. The 'Authorization' tab is selected, showing a 'Type' dropdown set to 'No Auth'. Below this, the 'Body' tab is selected, showing the response body in 'Pretty' format: `{"id":1,"title":"test","content":"lorem ipsum","created_at":null,"updated_at":null}`.

http://simple-rest-api/lum +

GET `http://simple-rest-api/lumen/public/api/article/1`

Authorization Headers Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (6) Tests

Pretty Raw Preview

```
{"id":1,"title":"test","content":"lorem ipsum","created_at":null,"updated_at":null}
```

Lumen от Laravel. Проверка

The screenshot displays a REST client interface with the following components:

- URL Bar:** Shows the base URL `http://simple-rest-api/lumen` and an environment dropdown set to `No environment`.
- Request Configuration:**
 - Method:** `POST`
 - URL:** `http://simple-rest-api/lumen/public/api/article?title=Hello!&content=loremIpsum`
 - Params:** A button to view query parameters.
 - Buttons:** `Send` (blue) and `Save` (grey).
- Request Body:**
 - Authorization:** A dropdown menu set to `No Auth`.
 - Headers:** A tab labeled `Headers (6)` is active.
- Response:**
 - Status:** `200 OK`
 - Time:** `1071 ms`
 - Body:** A tab labeled `Body` is active, showing the response in `Pretty` format:

```
{"title":"Hello!","content":"loremIpsum","updated_at":"2016-04-18 11:04:05","created_at":"2016-04-18 11:04:05","id":3}
```