



Construa uma API
em PHP com o
Laravel em pouco
tempo



Diego R. Mengarda

@diegomengarda

Desenvolvedor web há 8 anos;

Ex-professor IFSul Bagé;

Desenvolvedor de Software Sênior na

adopets

Iniciando nosso projeto

```
git clone https://github.com/diegomengarda/saci2019-api-start.git
```

```
cd saci2019-api/
```

```
./composer.phar create-project --prefer-dist laravel/laravel api "5.8.*"
```

```
cd api && touch database/database.sqlite
```

Iniciando nosso projeto / explicando...

```
# Clonar o repositório do git
```

```
git clone https://github.com/diegomengarda/saci2019-api.git
```

```
# Entrar na pasta criada após o git clone
```

```
cd saci2019-api/
```

```
# Rodar o comando que cria um projeto laravel na versão 5.8
```

```
./composer.phar create-project --prefer-dist laravel/laravel api "5.8.*"
```

```
# Rodar o comando para criamos o nosso arquivo do banco de dados
```

```
cd api && touch database/database.sqlite
```












O que vamos abordar















- PHP/Laravel 5.8;
- Conceitos de aplicações web;
- Comunicação entre cliente -> servidor;
- Arquitetura MVC.












O que vamos abordar















- API Rest (GET, POST, PUT, DELETE);
- Composer;
- JSON;
- *Autenticação com JWT;

Estrutura de pastas

- ▼  api
 - >  app
 - >  bootstrap
 - >  config
 - >  database
 - >  public
 - >  resources
 - >  routes
 - >  storage
 - >  tests
 - >  vendor

-  .editorconfig
-  .env
-  .env.example
-  .gitattributes
-  .gitignore
-  .styleci.yml
-  artisan
-  composer.json
-  composer.lock
-  package.json
-  phpunit.xml
-  readme.md
-  server.php
-  webpack.mix.js

- ▼  api
 - >  app
 - >  bootstrap
 - >  config
 - >  database
 - >  public
 - >  resources
 - >  routes
 - >  storage
 - >  tests
 - >  vendor

-  .editorconfig
-  .env
-  .env.example
-  .gitattributes
-  .gitignore
-  .styleci.yml
-  artisan
-  composer.json
-  composer.lock
-  package.json
-  phpunit.xml
-  readme.md
-  server.php
-  webpack.mix.js

Explicando o Laravel

- Models;
- Controllers;
- Migrations;
- Routes.

Criando o Model, Migration e Controller

Comando para criar o Model e a Migration

```
php artisan make:model Student --migration
```

Comando para criar o Controller

A flag --api serve para que o controller já seja criado com os métodos base para uma API Rest

```
php artisan make:controller StudentController --api
```

Model

StudentController.php × Student.php ×

```
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Student extends Model
8  {
9
10 }
```

```
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Student extends Model
8  {
9      protected $fillable = [
10         'first_name',
11         'last_name',
12         'nickname',
13         'age',
14         'instagram',
15         'facebook'
16     ];
17 }
18 |
```

Controller

```
6 use Illuminate\Http\Request;
7
8 class StudentController extends Controller
9 {
10     public function index()
11     {
12         return response([]);
13     }
14
15     public function store(Request $request)
16     {
17     }
18
19     public function show(int $studentId)
20     {
21     }
22
23     public function update(Request $request, int $studentId)
24     {
25     }
26
27     public function destroy(int $studentId)
28     {
29     }
```



```
public function index()  
{  
    $students = Student::all();  
    return response($students);  
}
```

```
public function store(Request $request)
{
    $data = $request→all();
    $student = new Student();
    $student→fill($data);
    $student→save();
    return response($student);
}
```

```
public function show(int $studentId)
{
    $student = Student::find($studentId);
    if (!$student) {
        return response(['error' => 'Student not found'], 400);
    }
    return response($student);
}
```

```
public function update(Request $request, int $studentId)
{
    $student = Student::find($studentId);
    if (!$student) {
        return response(['error' => 'Student not found'], 400);
    }

    $data = $request->all();
    $student->fill($data);
    $student->save();

    return response($student);
}
```

```
public function destroy(int $studentId)
{
    $student = Student::find($studentId);
    if (!$student) {
        return response(['error' => 'Student not found'], 400);
    }
    $student->delete();
    return response(['message' => 'Student deleted']);
}
```

Migration

```
public function up()  
{  
    Schema::create('students', function (Blueprint $table) {  
        $table->bigIncrements('id');  
        $table->timestamps();  
    });  
}
```

```
public function up()
{
    Schema::create('students', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string('first_name');
        $table->string('last_name');
        $table->string('nickname')->nullable();
        $table->smallInteger('age');
        $table->string('instagram')->nullable();
        $table->string('facebook')->nullable();
        $table->timestamps();
    });
}
```


Routes

```
15
16  Route::middleware('auth:api')→get('/user', function (Request $request) {
17     return $request→user();
18  });
19 |
```

```
16 // Lista todos estudantes
17 Route::get('student', 'StudentController@index');
18
19 // Cria um novo estudante
20 Route::post('student', 'StudentController@store');
21
22 // Busca dos dados de um estudante
23 Route::get('student/{studentId}', 'StudentController@show');
24
25 // Atualiza os dados de um estudante
26 Route::put('student/{studentId}', 'StudentController@update');
27
28 // Remove um estudante
29 Route::delete('student/{studentId}', 'StudentController@destroy');
30
```

Referências

<https://laravel.com/docs/5.8>

<https://medium.com/tableless/entendendo-tokens-jwt-json-web-token-413c6d1397f6>

Repo Principal: <https://github.com/diegomengarda/saci2019-api-start>

Repo API: <https://github.com/diegomengarda/saci2019-api>

Repo API - Auth JWT: <https://github.com/diegomengarda/saci2019-api-auth>

Extra

Autenticação utilizando o **JWT**

Obrigado!

Github: **diegomengarda**

E-mail: **diegormengarda@gmail.com**