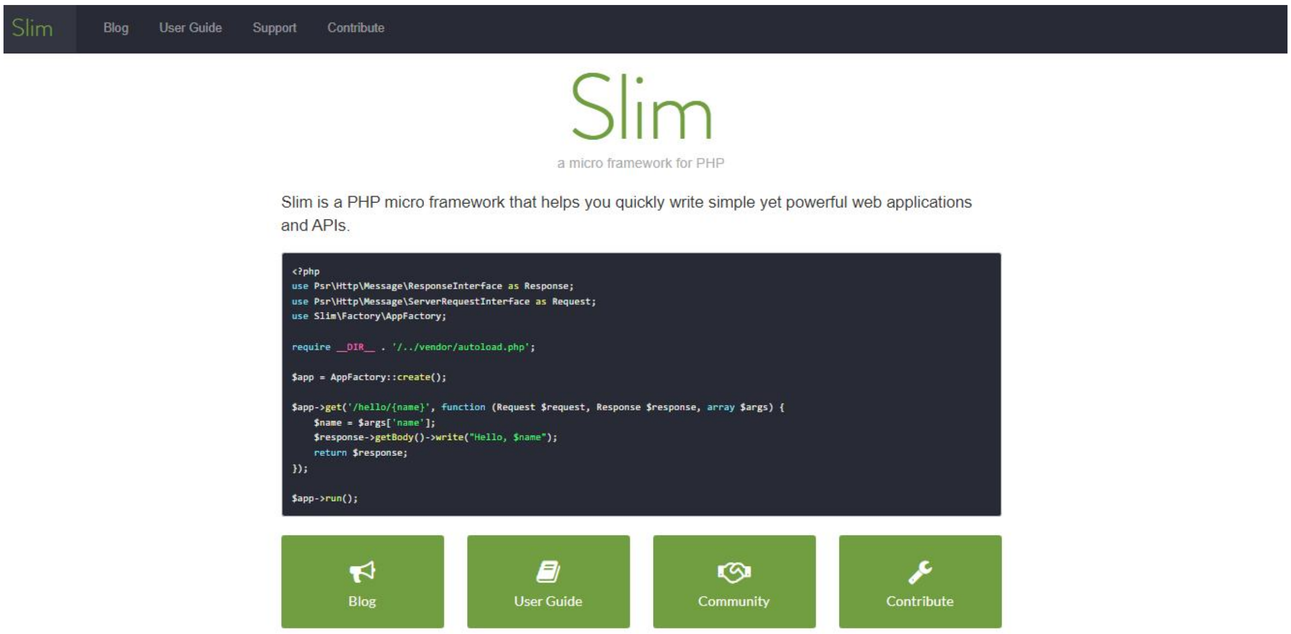
# Como criar uma API com o framework Slim
## Rafael Wendel Pinheiro

**Vídeo**: https://www.youtube.com/watch?v=WTf0lvJ0PLg

https://www.slimframework.com/

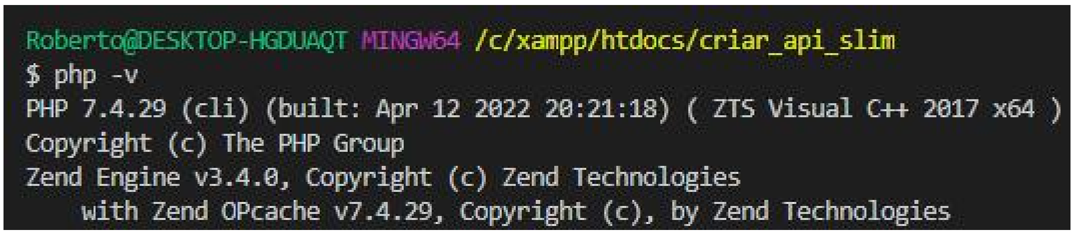


Slim is a PHP micro framework that helps you quickly write simple yet powerful web applications and APIs.

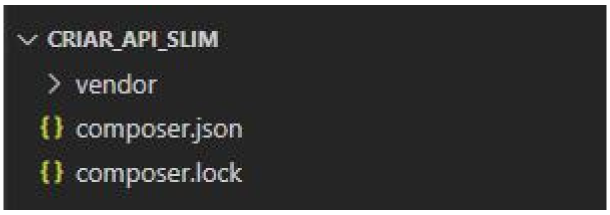
```php
<?php
use Psr\Http\Message\ResponseInterface as Response;
use Psr\Http\Message\ServerRequestInterface as Request;
use Slim\Factory\AppFactory;

require __DIR__ . '/../vendor/autoload.php';

$app = AppFactory::create();

$app->get('/hello/{name}', function (Request $request, Response $response, array $args) {
    $name = $args['name'];
    $response->getBody()->write("Hello, $name");
    return $response;
});

$app->run();
```

Versão mínima: PHP 7.2

php -v



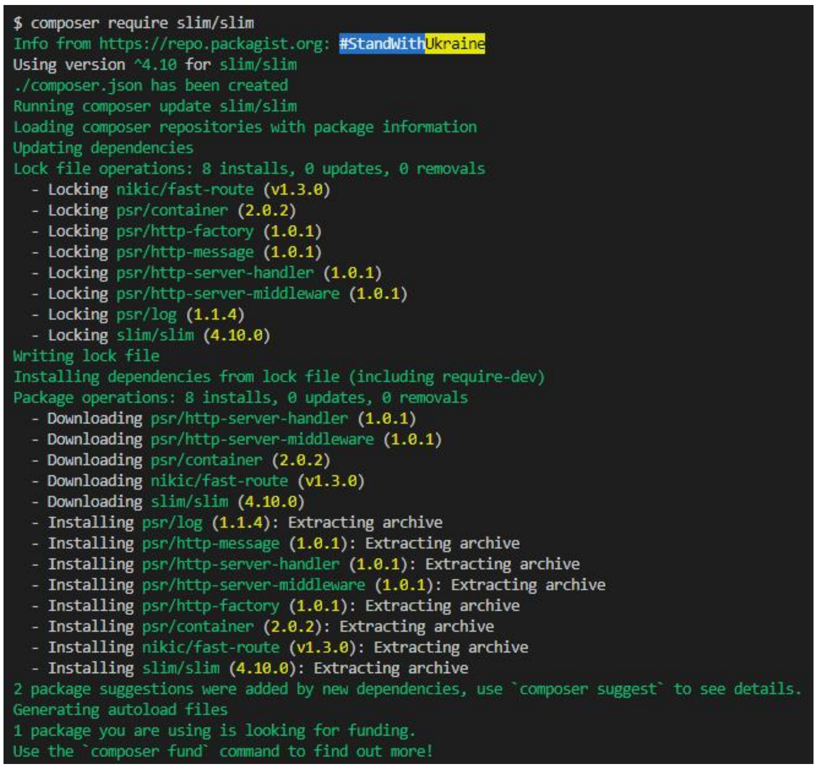
```
Roberto@DESKTOP-HGDUAQT MINGW64 /c/xampp/htdocs/criar_api_slim
$ php -v
PHP 7.4.29 (cli) (built: Apr 12 2022 20:21:18) ( ZTS Visual C++ 2017 x64 )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
    with Zend OPcache v7.4.29, Copyright (c), by Zend Technologies
```



```
CRIAR_API_SLIM
  > vendor
  {} composer.json
  {} composer.lock
```

## Instalando o framework Slim

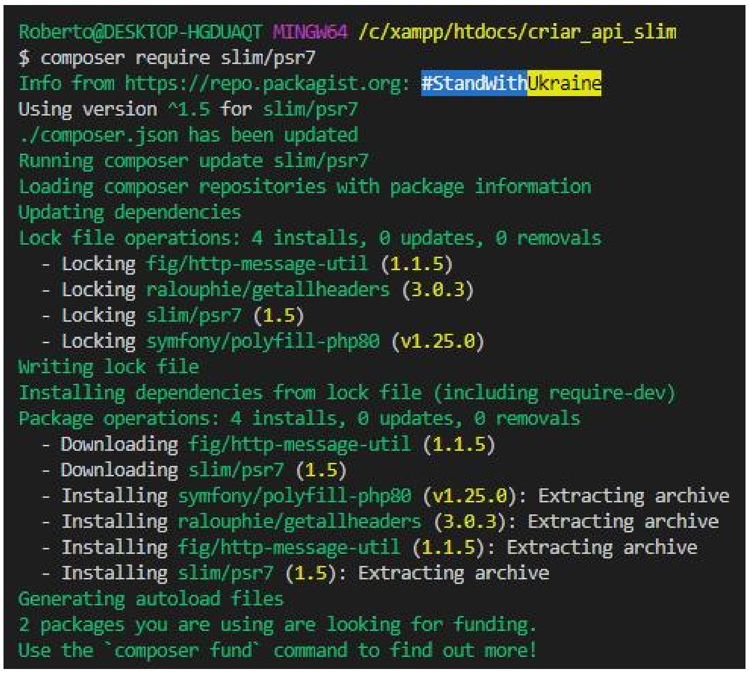
composer require slim/slim



```
$ composer require slim/slim
Info from https://repo.packagist.org: #StandWithUkraine
Using version ^4.10 for slim/slim
./composer.json has been created
Running composer update slim/slim
Loading composer repositories with package information
Updating dependencies
Lock file operations: 8 installs, 0 updates, 0 removals
  - Locking nikic/fast-route (v1.3.0)
  - Locking psr/container (2.0.2)
  - Locking psr/http-factory (1.0.1)
  - Locking psr/http-message (1.0.1)
  - Locking psr/http-server-handler (1.0.1)
  - Locking psr/http-server-middleware (1.0.1)
  - Locking psr/log (1.1.4)
  - Locking slim/slim (4.10.0)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 8 installs, 0 updates, 0 removals
  - Downloading psr/http-server-handler (1.0.1)
  - Downloading psr/http-server-middleware (1.0.1)
  - Downloading psr/container (2.0.2)
  - Downloading nikic/fast-route (v1.3.0)
  - Downloading slim/slim (4.10.0)
  - Installing psr/log (1.1.4): Extracting archive
  - Installing psr/http-message (1.0.1): Extracting archive
  - Installing psr/http-server-handler (1.0.1): Extracting archive
  - Installing psr/http-server-middleware (1.0.1): Extracting archive
  - Installing psr/http-factory (1.0.1): Extracting archive
  - Installing psr/container (2.0.2): Extracting archive
  - Installing nikic/fast-route (v1.3.0): Extracting archive
  - Installing slim/slim (4.10.0): Extracting archive
2 package suggestions were added by new dependencies, use `composer suggest` to see details.
Generating autoload files
1 package you are using is looking for funding.
Use the `composer fund` command to find out more!
```

## Baixando e instalando o Slim PSR-7

composer require slim/psr7



```
Roberto@DESKTOP-HGDUAQT MINGW64 /c/xampp/htdocs/criar_api_slim
$ composer require slim/psr7
Info from https://repo.packagist.org: #StandWithUkraine
Using version ^1.5 for slim/psr7
./composer.json has been updated
Running composer update slim/psr7
Loading composer repositories with package information
Updating dependencies
Lock file operations: 4 installs, 0 updates, 0 removals
  - Locking fig/http-message-util (1.1.5)
  - Locking ralouphie/getallheaders (3.0.3)
  - Locking slim/psr7 (1.5)
  - Locking symfony/polyfill-php80 (v1.25.0)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 4 installs, 0 updates, 0 removals
  - Downloading fig/http-message-util (1.1.5)
  - Downloading slim/psr7 (1.5)
  - Installing symfony/polyfill-php80 (v1.25.0): Extracting archive
  - Installing ralouphie/getallheaders (3.0.3): Extracting archive
  - Installing fig/http-message-util (1.1.5): Extracting archive
  - Installing slim/psr7 (1.5): Extracting archive
Generating autoload files
2 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

**.htaccess**

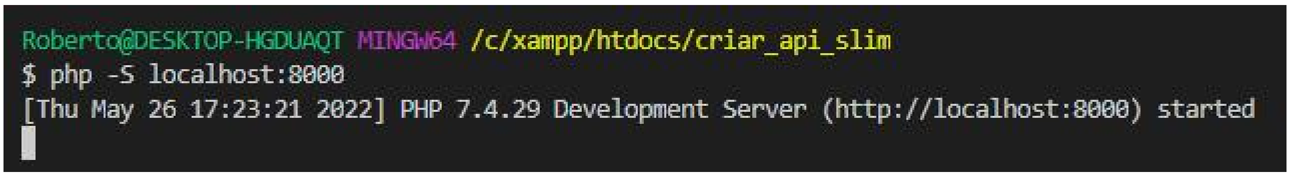
```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
Rewrite ^ index.php [QSA,L]
```

**index.php**

```
<?php
use Psr\Http\Message\ResponseInterface as Response;
use Psr\Http\Message\ServerRequestInterface as Request;
use Slim\Factory\AppFactory;

require __DIR__ . '/vendor/autoload.php';

$app = AppFactory::create();

$app->get('/', function (Request $request, Response $response, $args) {
    $response->getBody()->write("Hello world!");
    return $response;
});

$app->run();
```

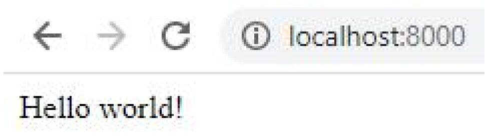
## Subindo o servidor PHP

php -S localhost:8000



```
Roberto@DESKTOP-HGDUAQT MINGW64 /c/xampp/htdocs/criar_api_slim
$ php -S localhost:8000
[Thu May 26 17:23:21 2022] PHP 7.4.29 Development Server (http://localhost:8000) started
```

localhost:8000

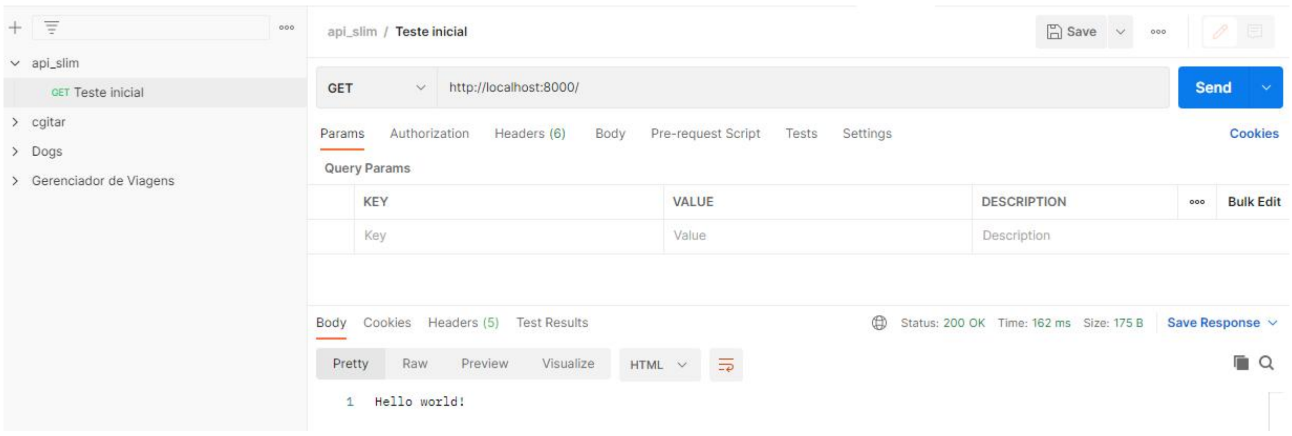


Hello world!

# Usando o Postman

## Teste inicial

GET
http://localhost:8000/



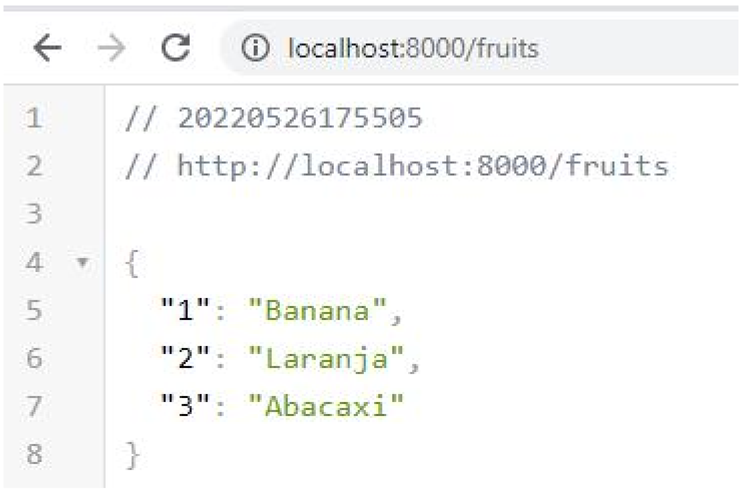
```
Hello world!
```

## Listando todas as frutas

**index.php**

```
<?php

use Psr\Http\Message\ResponseInterface as Response;
use Psr\Http\Message\ServerRequestInterface as Request;
use Slim\Factory\AppFactory;

require __DIR__ . '/vendor/autoload.php';

$app = AppFactory::create();

$app->get('/', function (Request $request, Response $response, $args) {
    $response->getBody()->write("Hello world!");
    return $response;
});

$app->get('/fruits', function (Request $request, Response $response, $args) {
    $fruits = [
        '1' => 'Banana',
        '2' => 'Laranja',
        '3' => 'Abacaxi'
    ];

    $response->getBody()->write(json_encode($fruits));
    return $response->withHeader('Content-Type', 'application/json');
});


$app->run();
```

**http://localhost:8000/fruits**



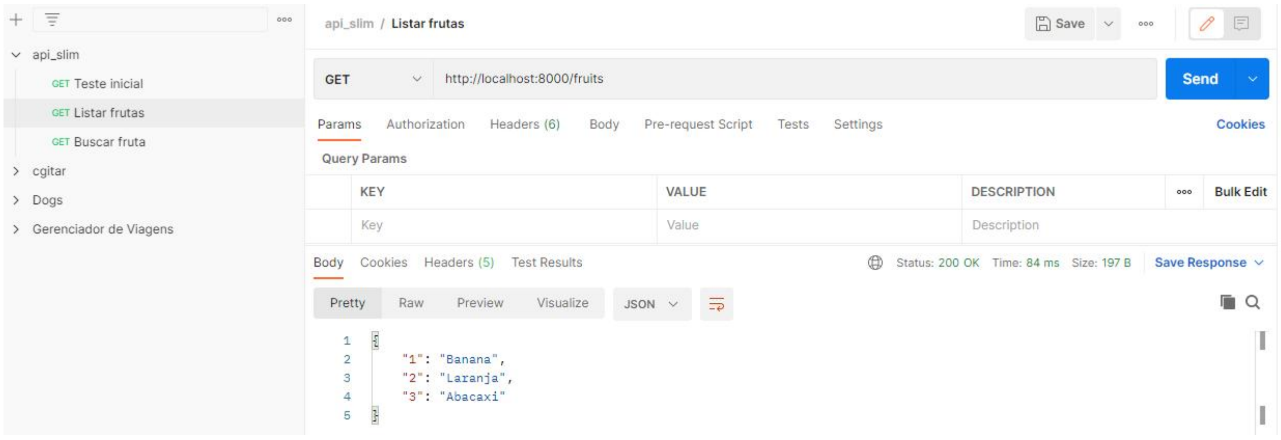
localhost:8000/fruits

```
// 20220526175505
// http://localhost:8000/fruits

{
  "1": "Banana",
  "2": "Laranja",
  "3": "Abacaxi"
}
```

- No Postman:

## Listando frutas

GET
http://localhost:8000/fruits



GET http://localhost:8000/fruits

Status: 200 OK Time: 84 ms Size: 197 B

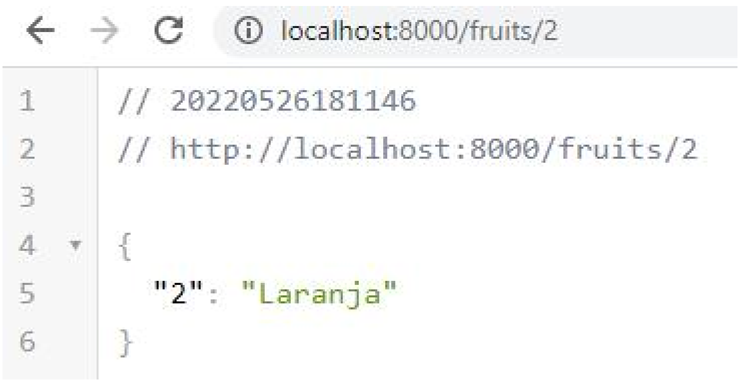
```
{
    "1": "Banana",
    "2": "Laranja",
    "3": "Abacaxi"
}
```

## Buscando uma fruta

**index.php**

```
<?php

use Psr\Http\Message\ResponseInterface as Response;
use Psr\Http\Message\ServerRequestInterface as Request;
use Slim\Factory\AppFactory;

require __DIR__ . '/vendor/autoload.php';

$app = AppFactory::create();

$app->get('/', function (Request $request, Response $response, $args) {
    $response->getBody()->write("Hello world!");
    return $response;
});

$app->get('/fruits', function (Request $request, Response $response, $args) {
    $fruits = [
        '1' => 'Banana',
        '2' => 'Laranja',
        '3' => 'Abacaxi'
    ];

    $response->getBody()->write(json_encode($fruits));
    return $response->withHeader('Content-Type', 'application/json');
});

$app->get('/fruits/{id}', function (Request $request, Response $response, $args) {
    $fruits = [
        '1' => 'Banana',
        '2' => 'Laranja',
        '3' => 'Abacaxi'
    ];

    $fruit[$args['id']] = $fruits[$args['id']];

    $response->getBody()->write(json_encode($fruit));
    return $response->withHeader('Content-Type', 'application/json');
});

$app->run();
```

http://localhost:8000/fruits/2

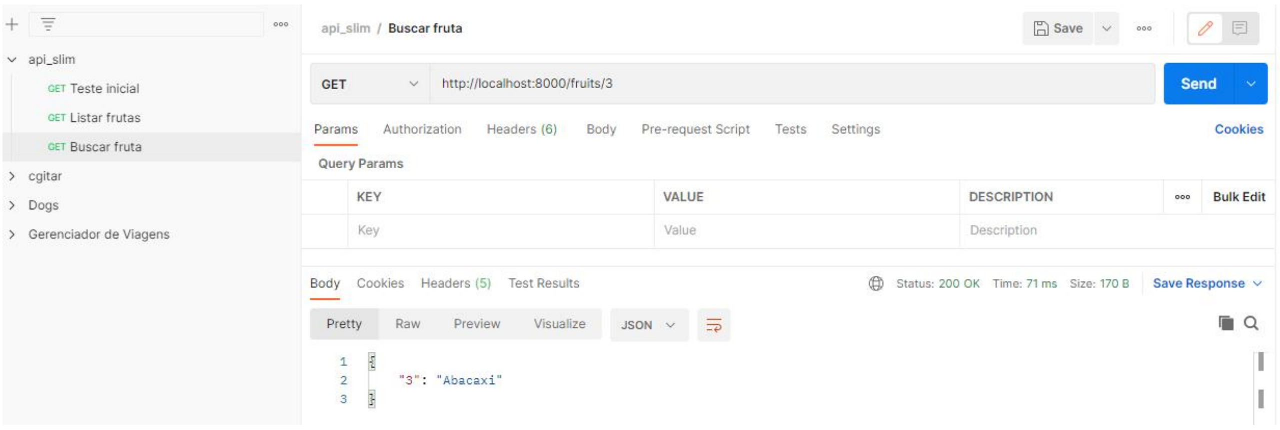


```
// 20220526181146
// http://localhost:8000/fruits/2

{
  "2": "Laranja"
}
```

No Postman:

## Buscar fruta

GET
http://localhost:8000/fruits/3



http://localhost:8000/fruits/3

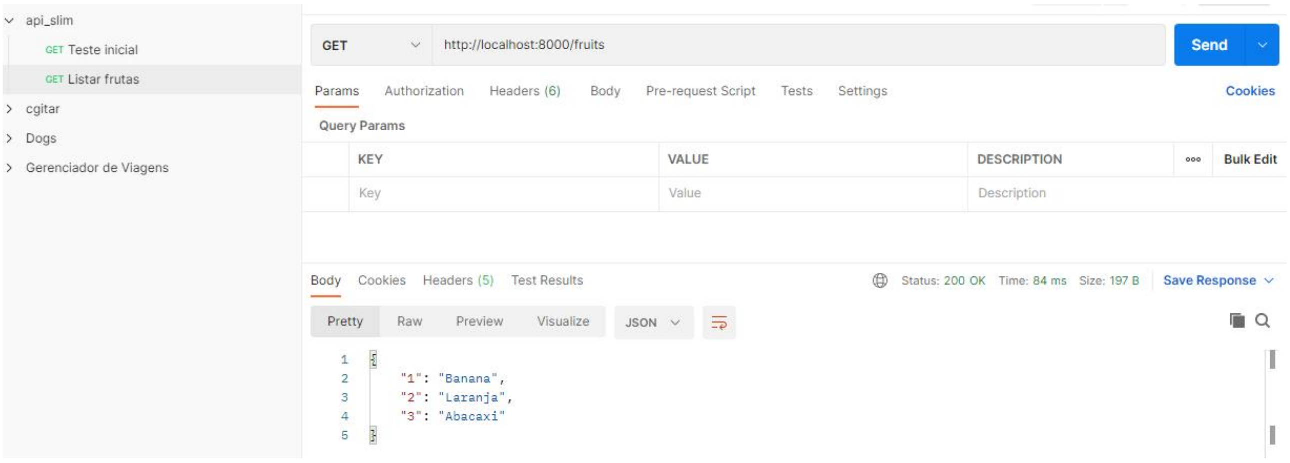
Status: 200 OK Time: 71 ms Size: 170 B

```
{
    "3": "Abacaxi"
}
```

- No Postman:

## Buscar fruta

GET
http://localhost:8000/fruits



http://localhost:8000/fruits

Status: 200 OK Time: 84 ms Size: 197 B

```
{
    "1": "Banana",
    "2": "Laranja",
    "3": "Abacaxi"
}
```