

```
string sInput;  
int iLength, iN;  
double dblTemp;  
bool again = true;
```

```
while (again) {  
    iN = -1;  
    again = false;  
    getline(cin, sInput);  
    system("cls");  
    stringstream(sInput) >> dblTemp;  
    iLength = sInput.length();  
    if (iLength < 4) {  
        again = true;  
        continue;  
    } else if (sInput[iLength - 3] != '.') {  
        again = true;  
        continue;  
    } while (++iN < iLength) {  
        if (isdigit(sInput[iN])) {  
            continue;  
        } else if (iN == (iLength - 3)) {  
            continue;  
        }  
    }  
}
```

LARAVEL

Module 02 Routage



Au programme dans ce module

- ❑ Chapitre 1 : Début et fin du routage
 - public/index.php
 - RouteServiceProvider.php
 - routes/web.php

- ❑ Chapitre 2 : Les bases du routage
 - Les verbes de routage
 - Personnalisation des routes
 - Les variables dans l'URL
 - Insérer des routes

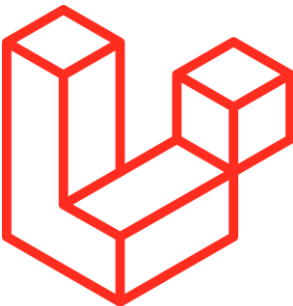
- ❑ Chapitre 3 : Mise en pratique



Début et fin du routage

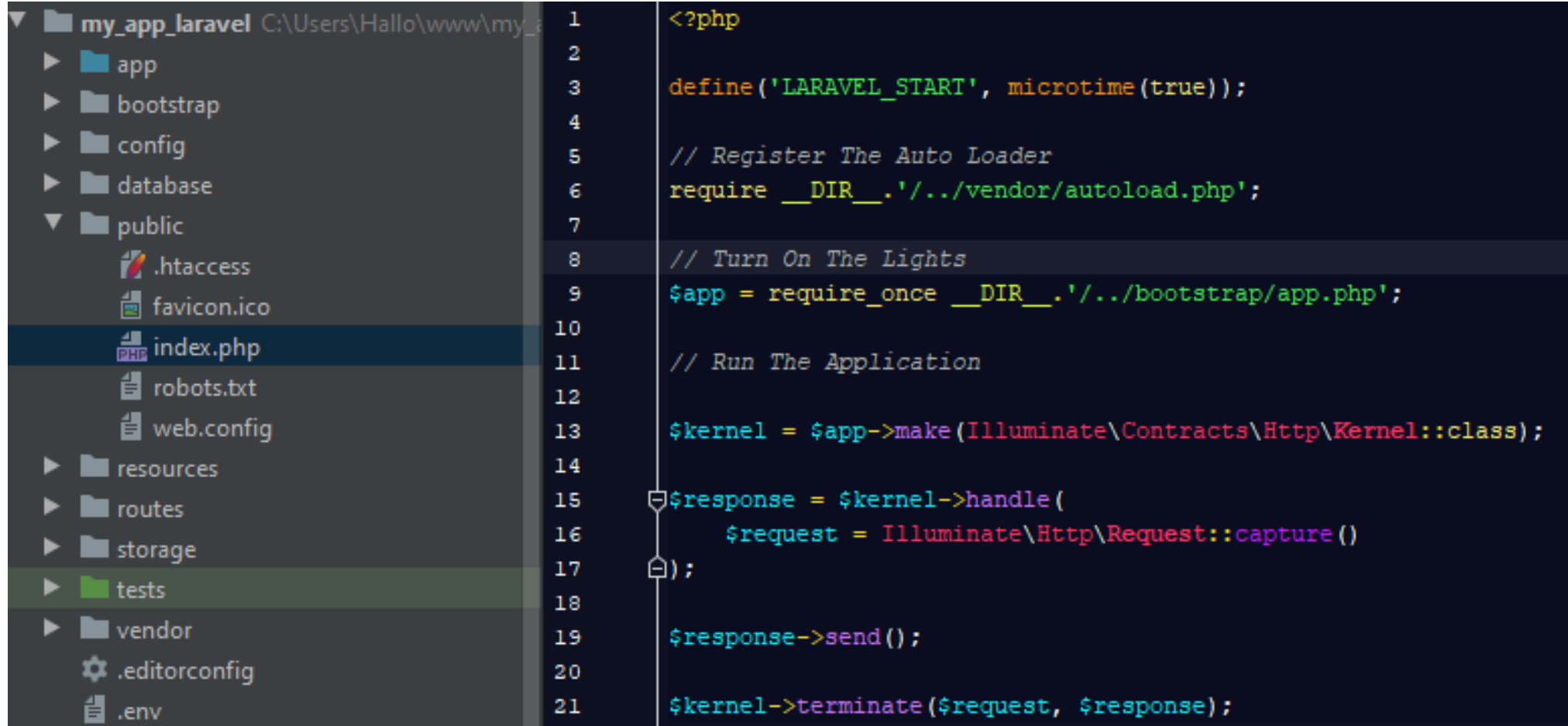


Chapitre 1



Début et fin du routage

public/index.php

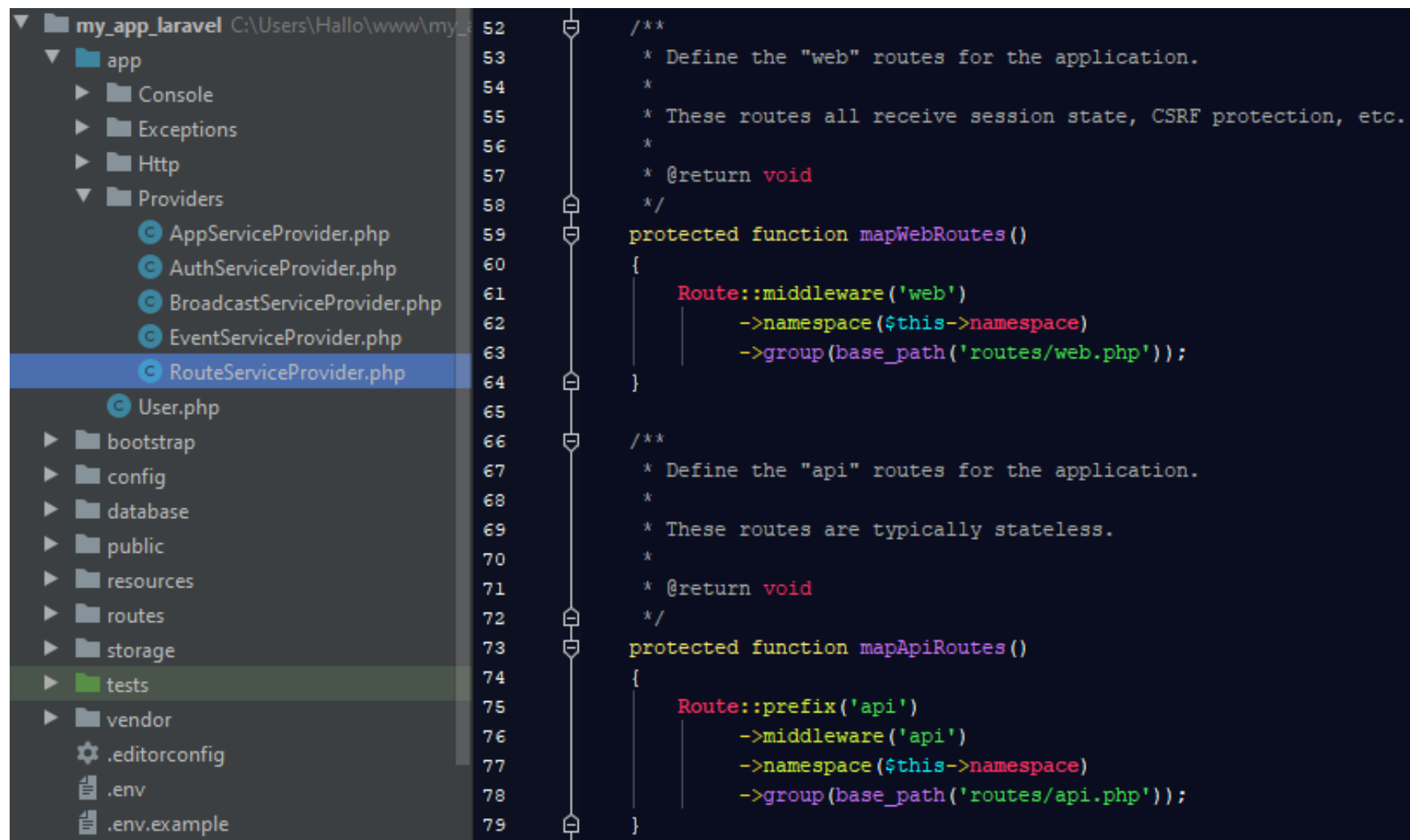


The image shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows the directory structure of a Laravel application, with the 'public' directory expanded and 'index.php' selected. The code editor shows the contents of 'public/index.php', which is a PHP script that initializes the Laravel application.

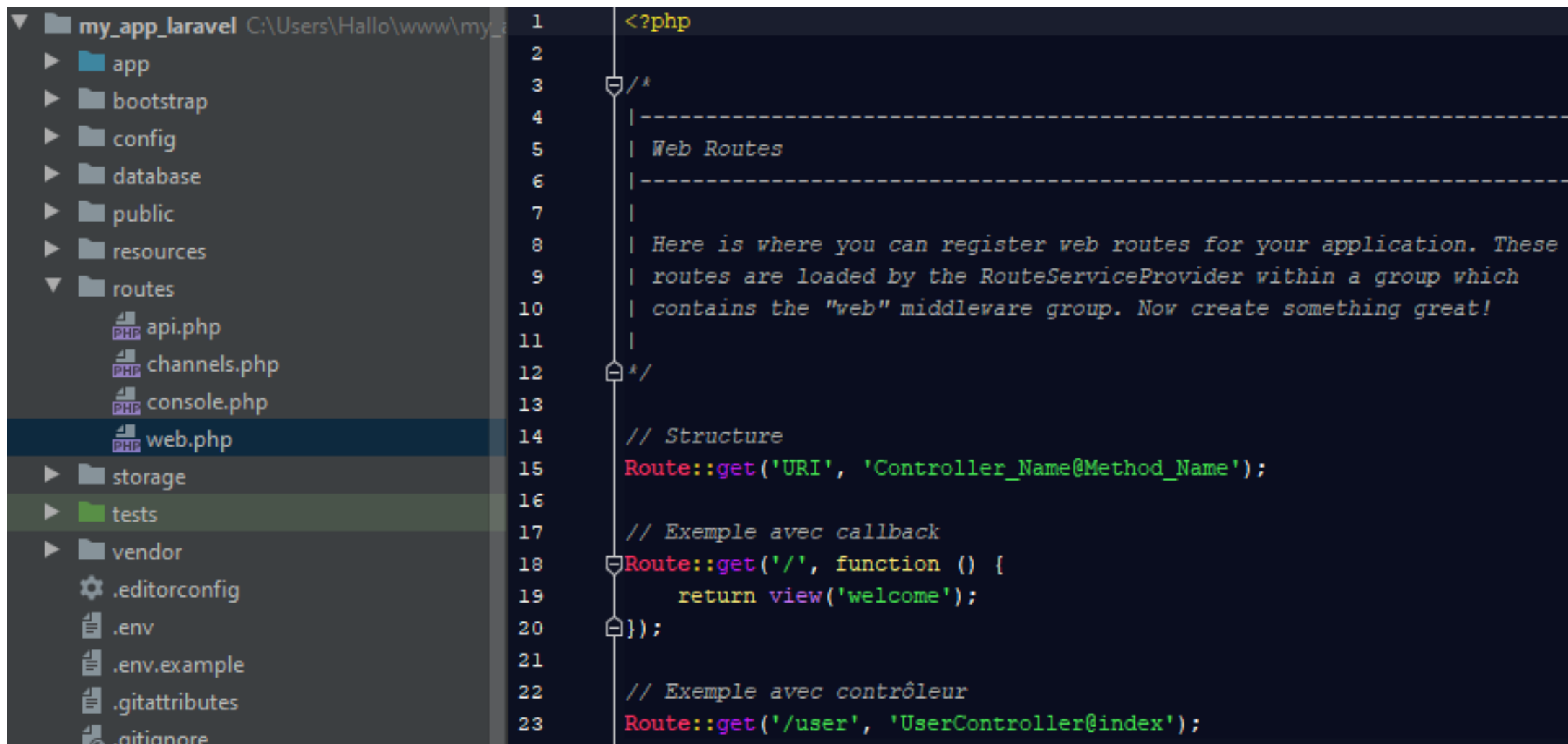
```
1 <?php
2
3 define('LARAVEL_START', microtime(true));
4
5 // Register The Auto Loader
6 require __DIR__.'/../vendor/autoload.php';
7
8 // Turn On The Lights
9 $app = require_once __DIR__.'/../bootstrap/app.php';
10
11 // Run The Application
12
13 $kernel = $app->make(Illuminate\Contracts\Http\Kernel::class);
14
15 $response = $kernel->handle(
16     $request = Illuminate\Http\Request::capture()
17 );
18
19 $response->send();
20
21 $kernel->terminate($request, $response);
```



RouteServiceProvider.php



routes/web.php



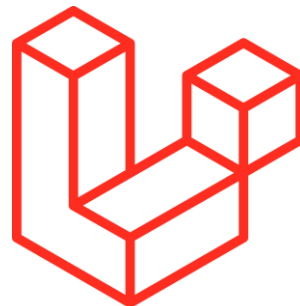
```
1 <?php
2
3 /*
4 |-----
5 | Web Routes
6 |-----
7 |
8 | Here is where you can register web routes for your application. These
9 | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 // Structure
15 Route::get('URI', 'Controller_Name@Method_Name');
16
17 // Exemple avec callback
18 Route::get('/', function () {
19     return view('welcome');
20 });
21
22 // Exemple avec contrôleur
23 Route::get('/user', 'UserController@index');
```



Les bases du routage



Chapitre 2



Les verbes de routage

```

- /*
  |-----
  | Web Routes
  |-----
- */

// Verbe HTTP
Route::get($uri, $callback);
Route::post($uri, $callback);
Route::put($uri, $callback);
Route::patch($uri, $callback);
Route::delete($uri, $callback);
Route::options($uri, $callback);

// Match only GET and POST
Route::match(['get', 'post'], $uri, $callback);
// Match all verbs
Route::any($uri, $callback);
```



Personnalisation des routes

```
// Nommage par méthode (name)
Route::get('/user', 'UserController@index')->name('user.index');
Route::get('/user', 'UserController@index')->name('user.get');
Route::get('/user', 'UserController@index')->name('user');

// Nommage par options (as et use)
Route::get('/user', ['as' => 'user.index', 'use' => 'UserController@index']);

// Groupe "admin" avec préfixe (prefix) et extension de nom (as)
Route::prefix('admin')->as('admin.')->group(function () {
    Route::get('/', ['as' => 'index', 'use' => 'AdminController@index']);
    // --> Match avec /admin
    Route::get('profile', ['as' => 'profile', 'use' => 'AdminController@profile']);
    // --> Match avec /admin/profile
});

// Équivaut à :
Route::get('admin', ['as' => 'admin.index', 'use' => 'AdminController@index']);
Route::get('admin/profile', ['as' => 'admin.profile', 'use' => 'AdminController@profile']);
```



Les variables dans l'URL

```
// Variable simple obligatoire
Route::get('user/{id}', function ($id) {
    return 'User ID : '.$id;
});
Route::get('user/{id}', 'UserController@show');

// Variable simple optionnelle
Route::get('user/{id?}', function ($id = -1) {
    return $id;
});

// Variable complexe avec nombre
Route::get('user/{id}', function ($id) {
    return $id;
})->where('id', '[0-9]+'); // Voir la doc pour toutes les variantes

// Variable complexe avec lettre
Route::get('user/{id}', function ($id) {
    return $id;
})->where('id', '[A-Za-z]+'); // Voir la doc pour toutes les variantes
```



Insérer des routes

```
// Route avec variable simple  
Route::get('user/{id}', ['as' => 'user.show', 'use' => 'UserController@show']);
```

```
// Dans une méthode de Contrôleur ou dans une Vue.  
route('user.show', [$id]);
```

Pour info, « route » est un helper. Une méthode importée par Laravel au boot de l'app et qui est disponible un peu partout comme toutes méthode native de PHP.

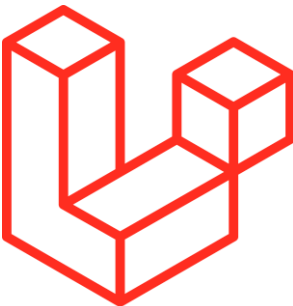
Plus de détails à <https://laravel.com/docs/6.x/routing>



Mise en pratique



Chapitre 3



Mise en pratique

- ❖ Téléchargez le logiciel : Postman
- ❖ Créez et testez les routes (uniquement avec des fonctions anonymes) :
 - "index" (GET)
 - "list" (GET)
 - "show" (GET) avec un paramètre optionnel
 - "store" (POST)
 - "update" (PUT) avec un paramètre en fin d'url
 - "delete" (DELETE) avec un paramètre en fin d'url et qui doit être uniquement des lettres



```
17 string sInput;  
18 int iLength, iN;  
19 double dblTemp;  
20 bool again = true;  
21  
22 while (again) {  
23     iN = -1;  
24     again = false;  
25     getline(cin, sInput);  
26     system("cls");  
27     stringstream(sInput) >> dblTemp;  
28     iLength = sInput.length();  
29     if (iLength < 4) {  
30         again = true;  
31         continue;  
32     } else if (sInput[iLength - 3] != '.') {  
33         again = true;  
34         continue;  
35     } while (++iN < iLength) {  
36         if (isdigit(sInput[iN])) {  
37             continue;  
38         } else if (iN == (iLength - 3)) {  
39             continue;  
40         }  
41     }  
42     // ...  
43 }
```

Fin du module

