# Objective = add a web-service

In this tutorial you will learn how to add a web-service that computes

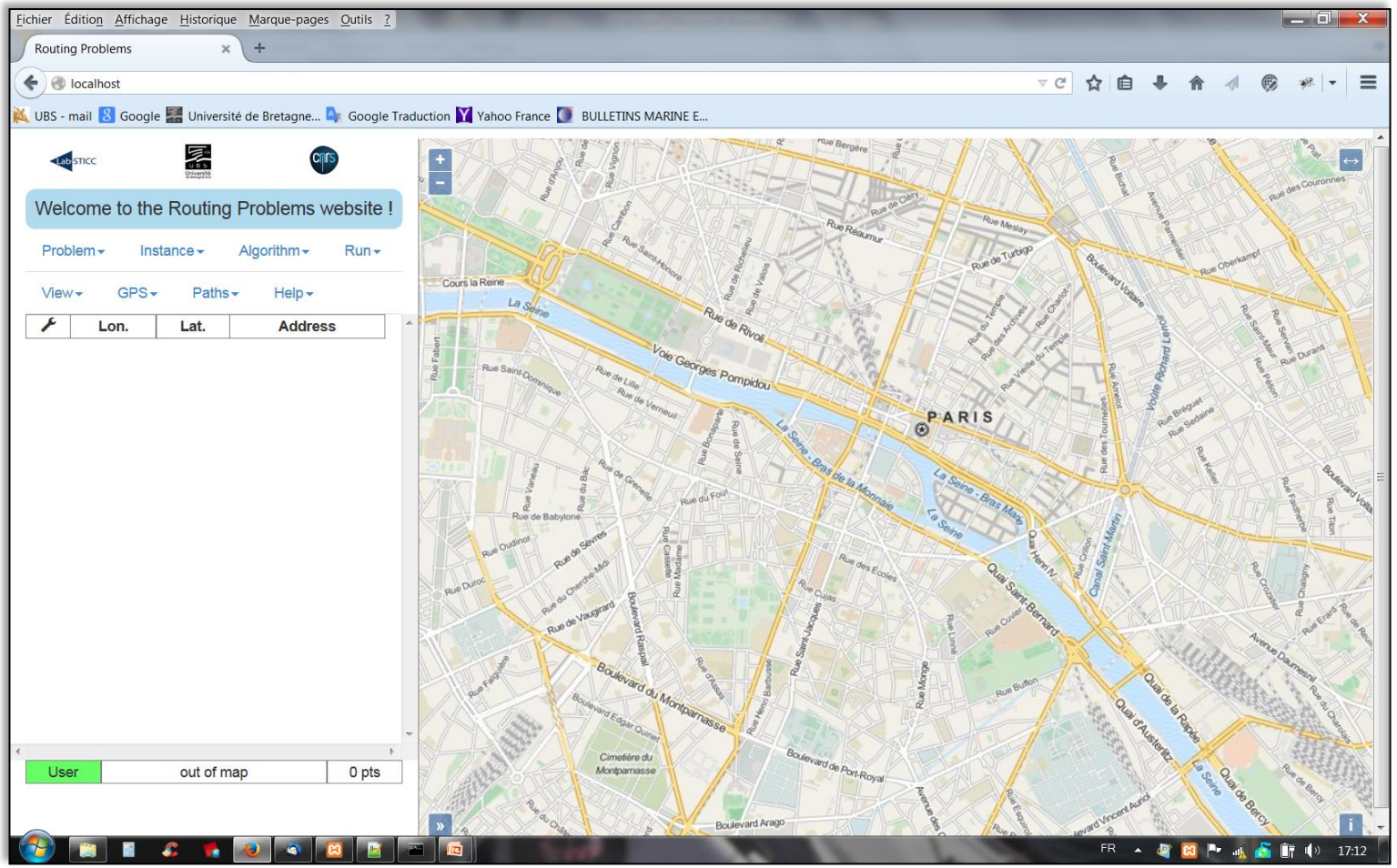
$$y = ax + b$$

We suppose you are using Firefox, Chrome or IE.

We suppose you have a running, and well-configured, apache server on your PC.

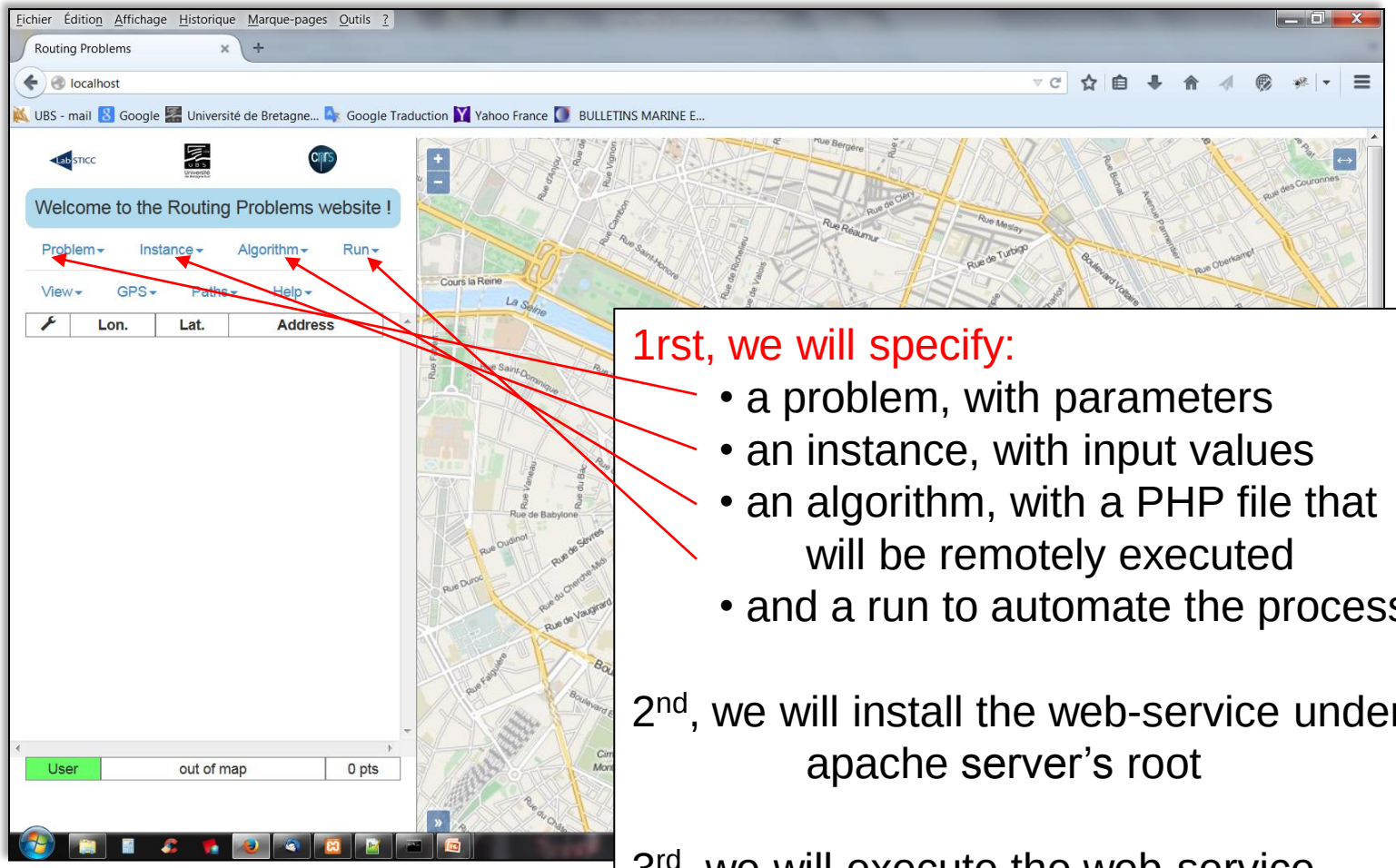
Pierre Bomel & Marc Sevaux
Université de Bretagne Sud,
Lab-STICC, Lorient, France, 2015

# Start a browser with the URL "localhost"




You should see exactly this.

# Start a browser with the URL "localhost"



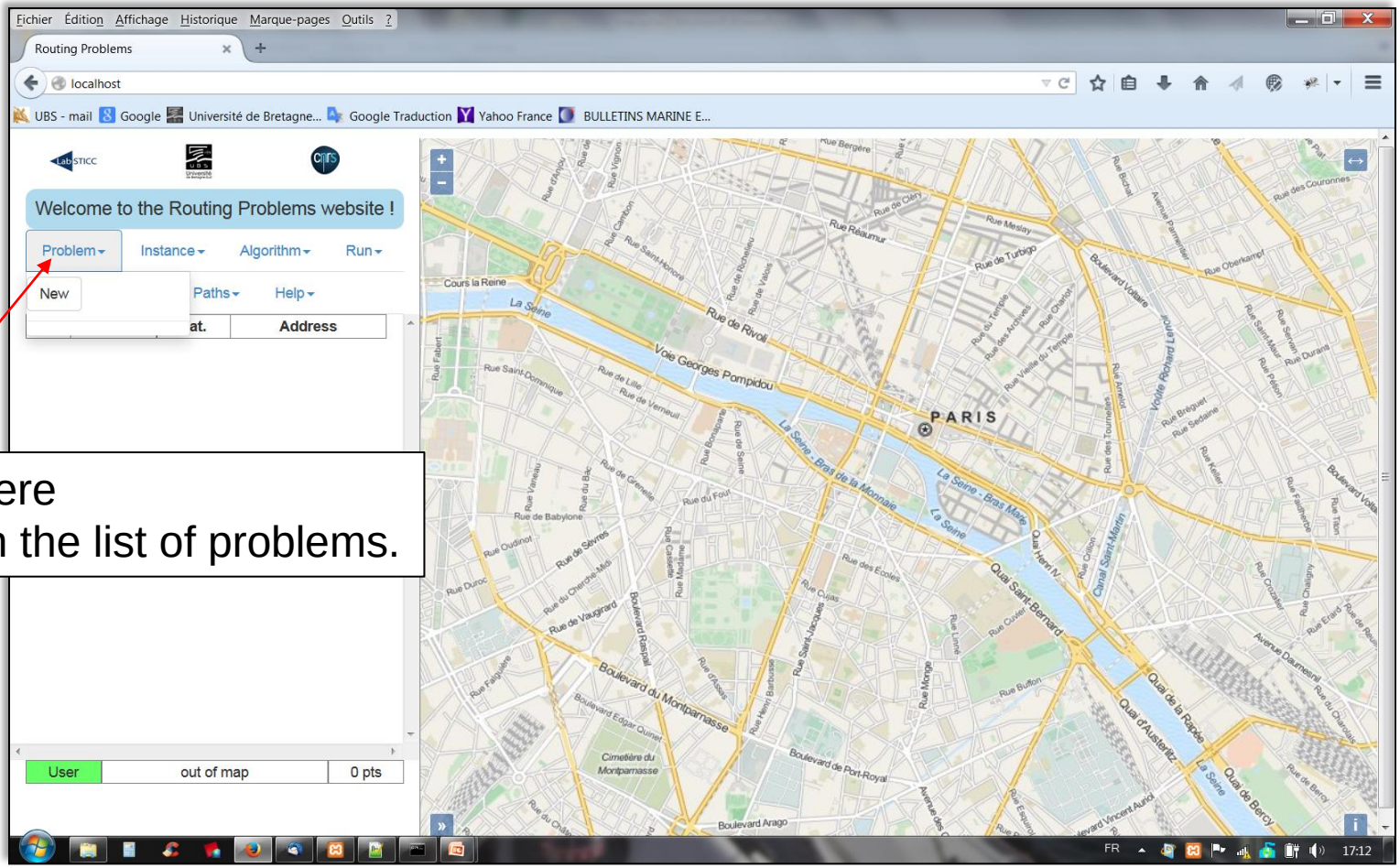

1rst, we will specify:

- a problem, with parameters
- an instance, with input values
- an algorithm, with a PHP file that will be remotely executed
- and a run to automate the process

2nd, we will install the web-service under the apache server's root

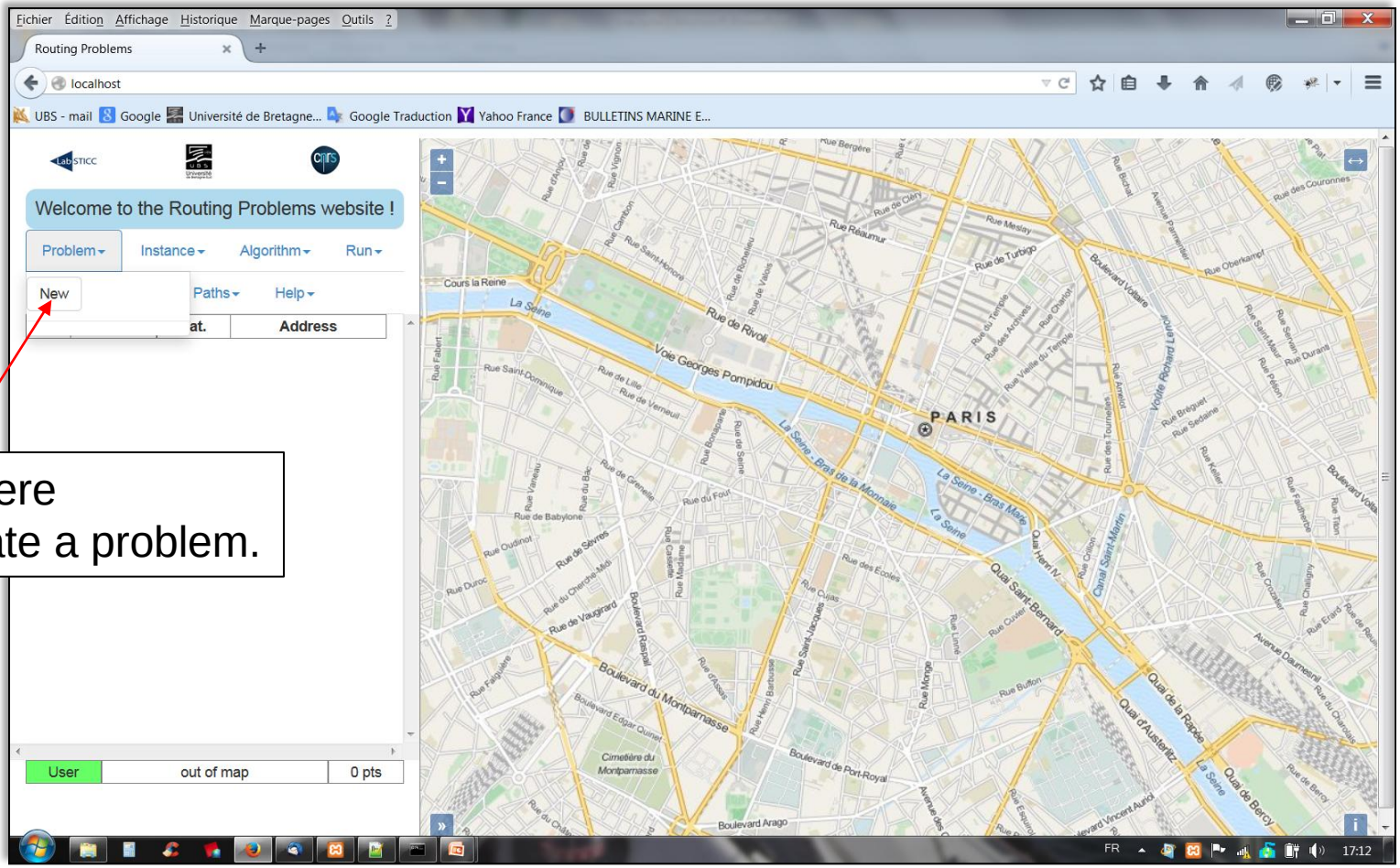
3rd, we will execute the web-service

# Specifications: open the list of problems.



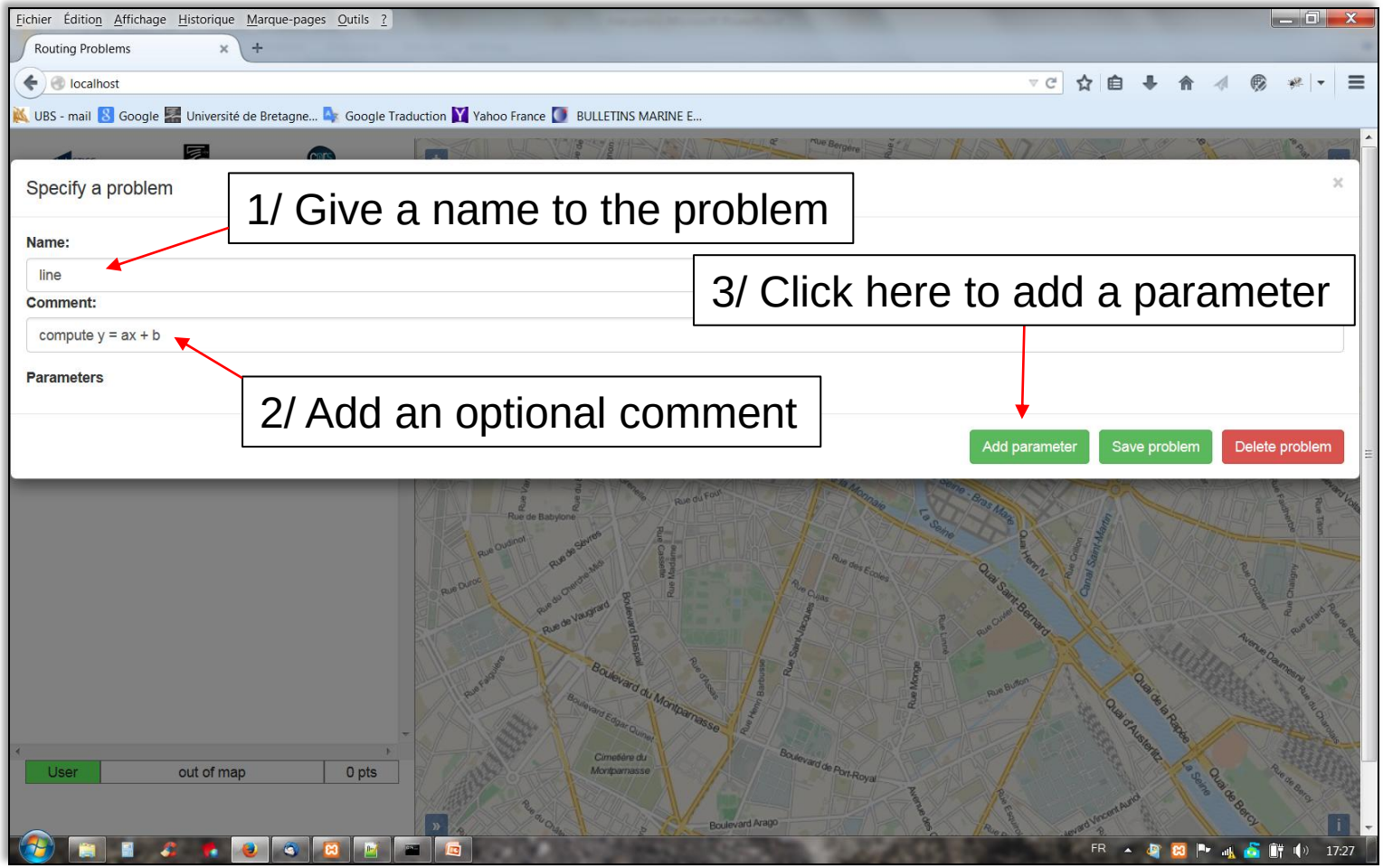

Click here
to open the list of problems.

# Specifications: create a problem.



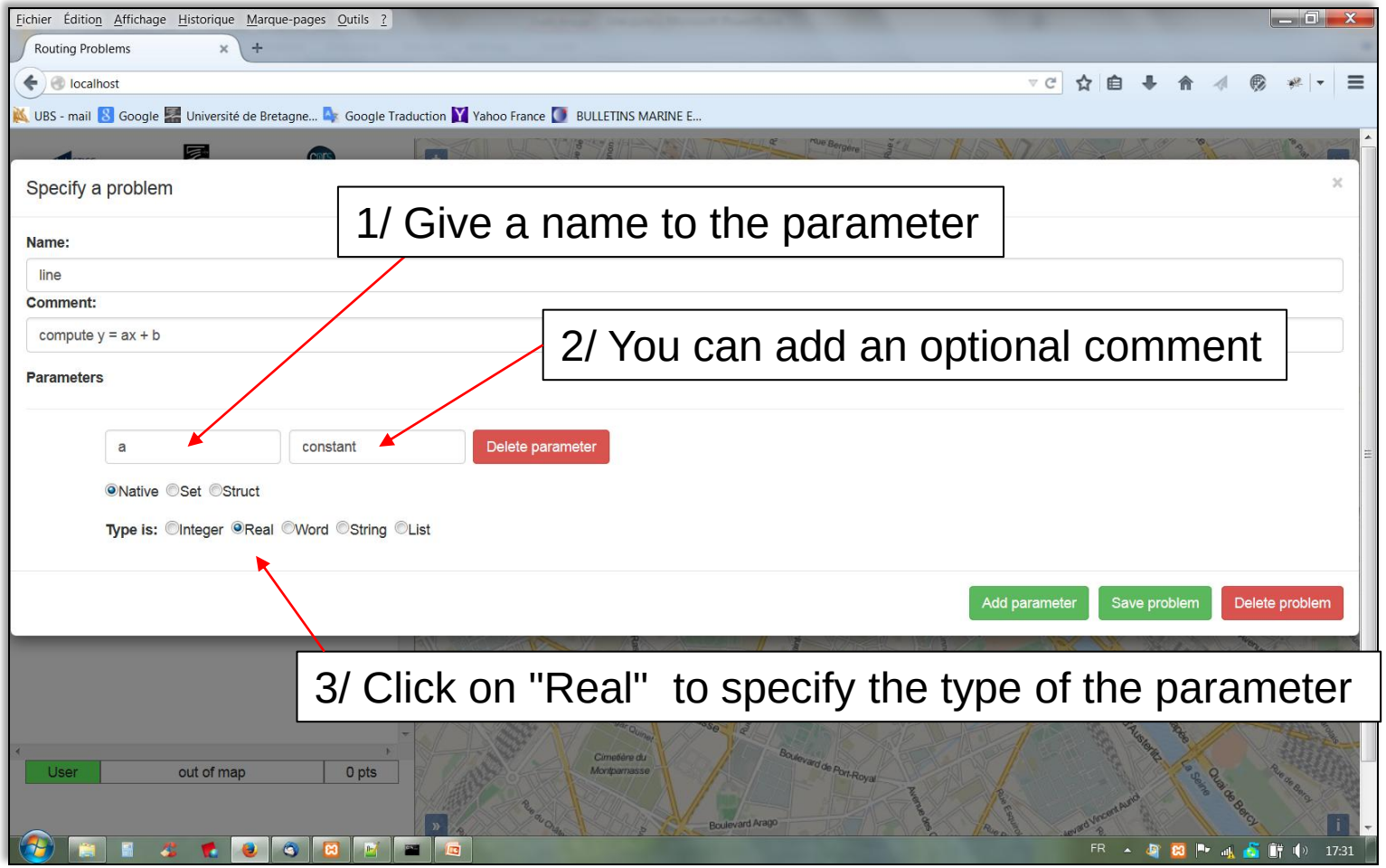

Click here
To create a problem.

# Specifications: fill the form for the problem and create parameters.

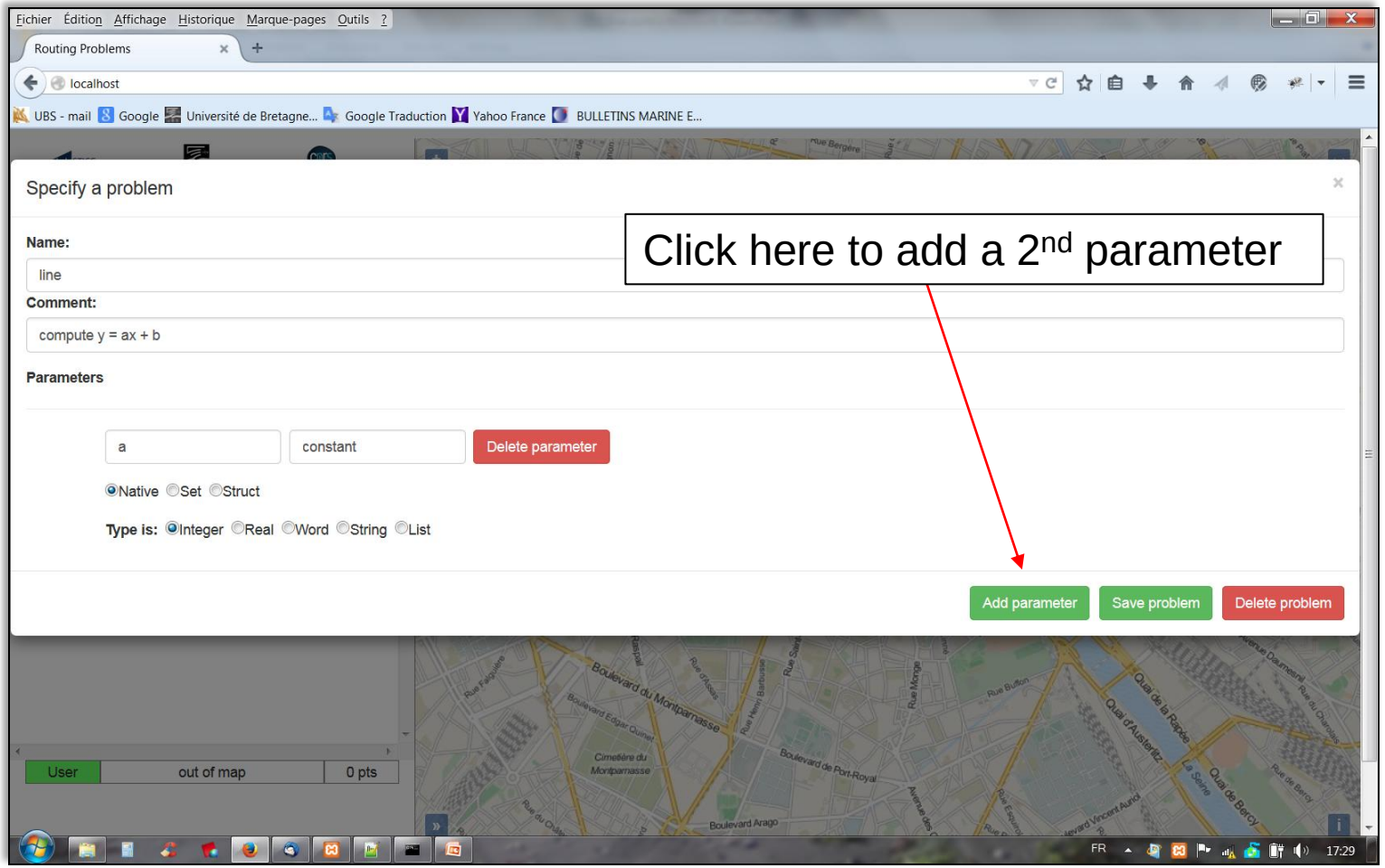



A window named « Specify a problem » opens.

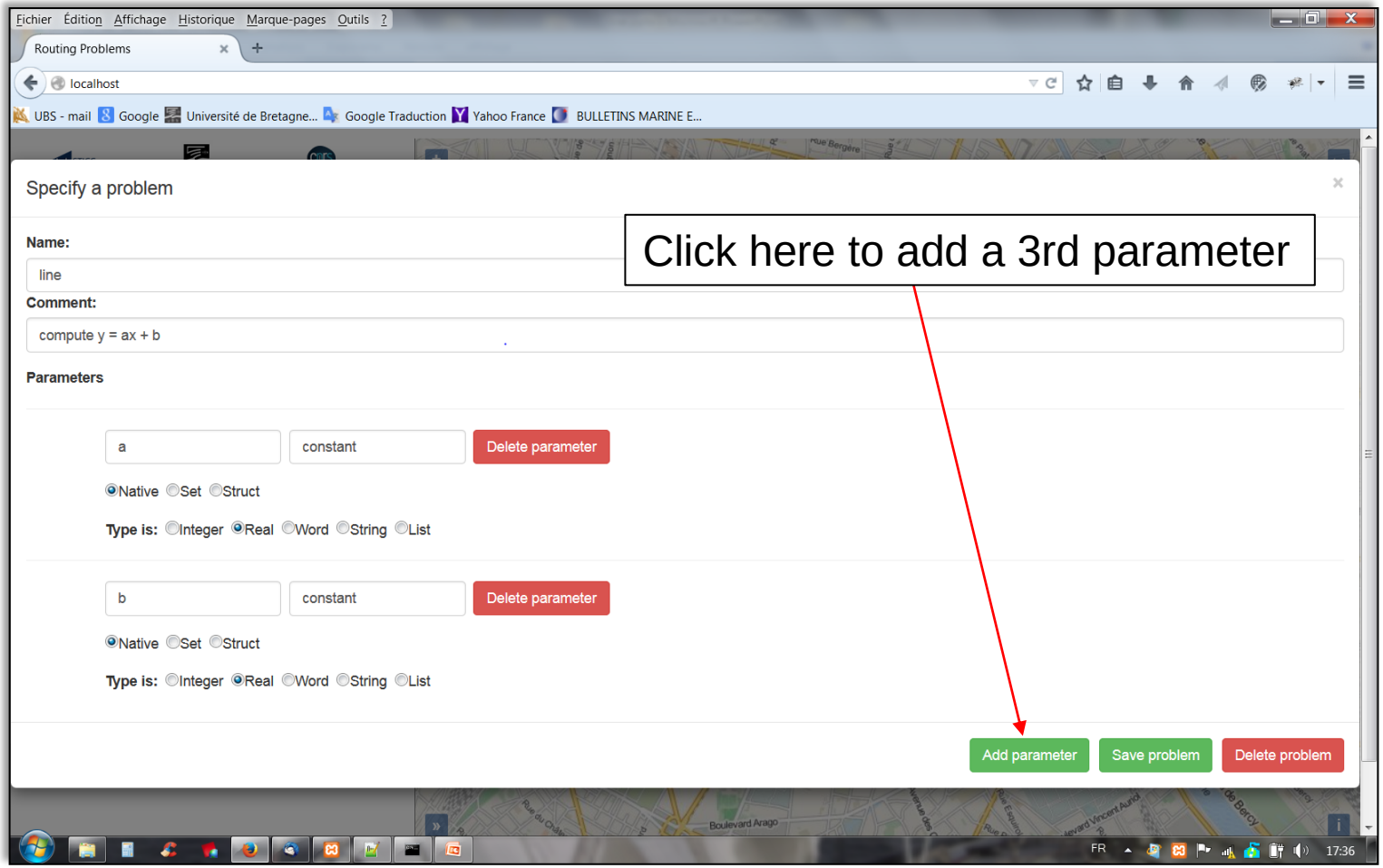
# Specifications: fill the form for the problem and create parameters.



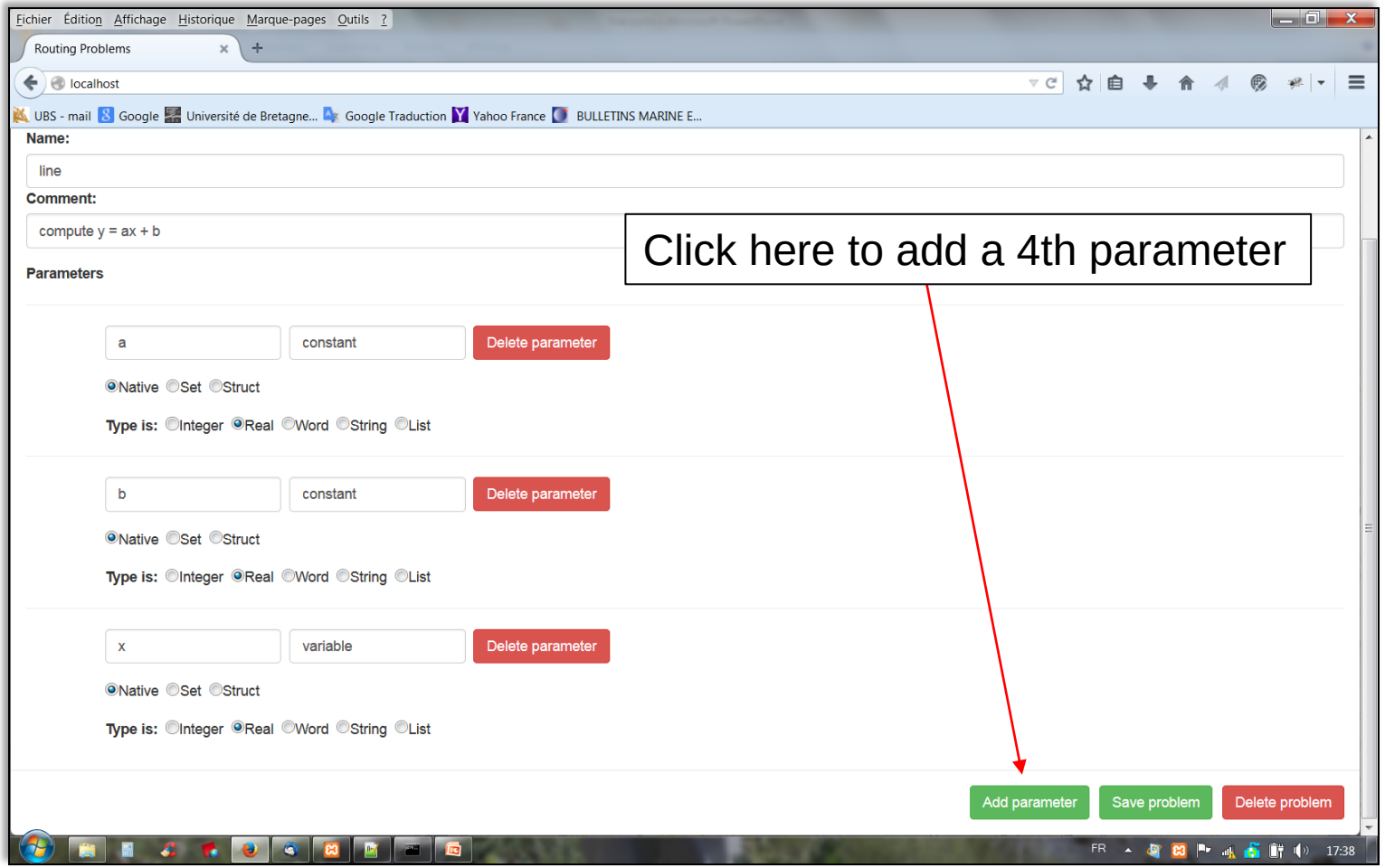
# Specifications: fill the form for the problem and create parameters.



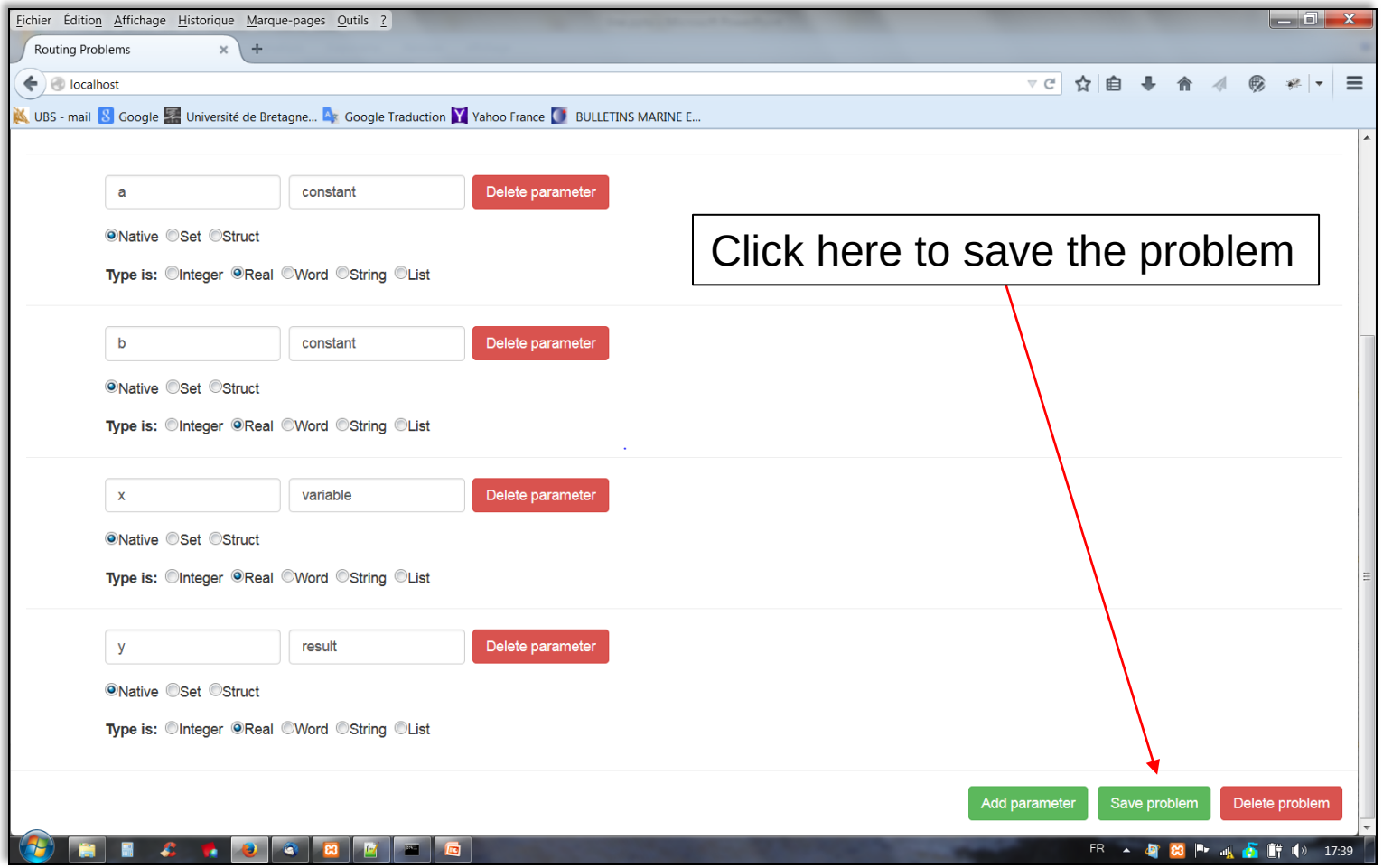
# Specifications: fill the form for the problem and create parameters.



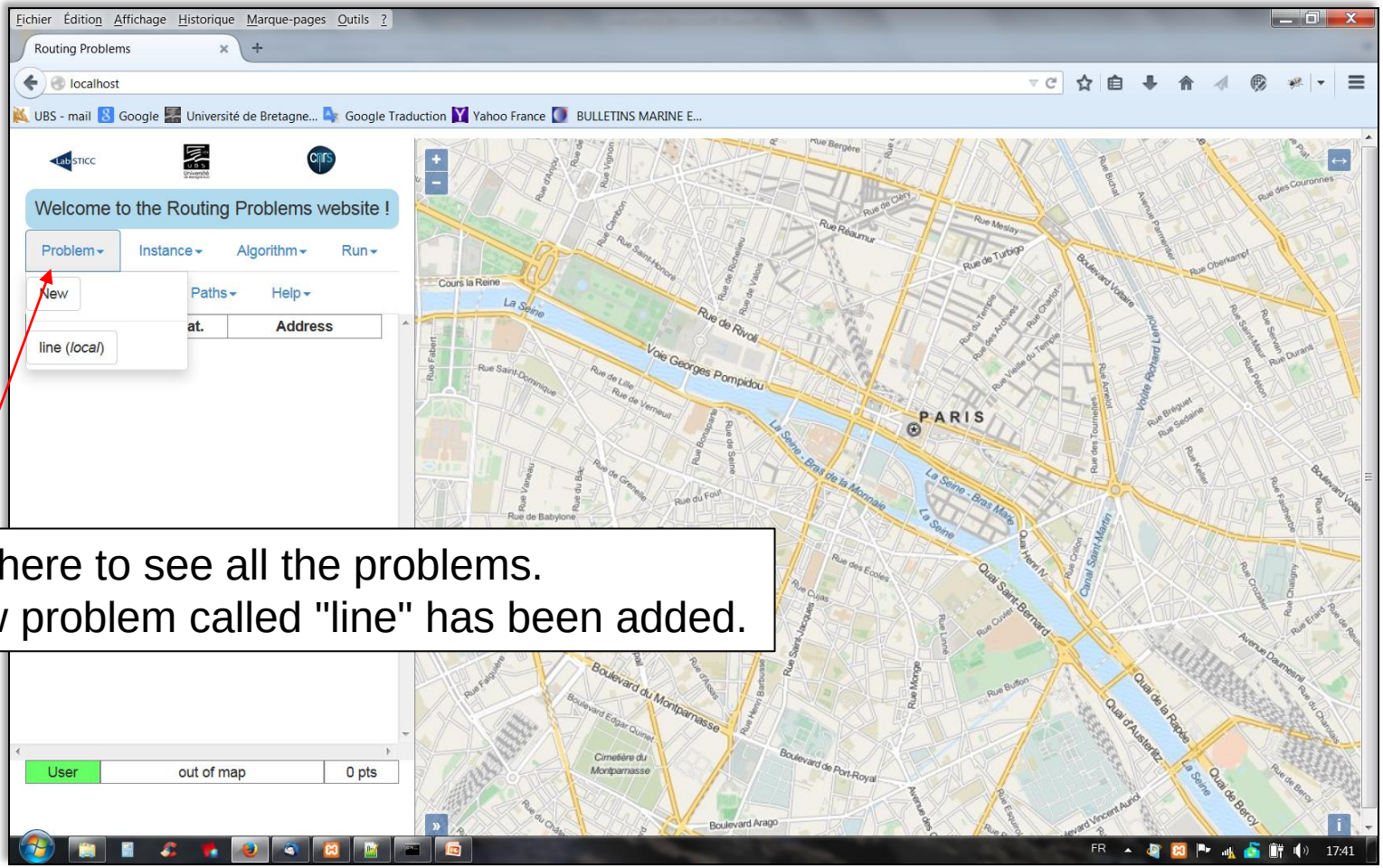
# Specifications: fill the form for the problem and create parameters.



# Specifications: fill the form for the problem and create parameters.

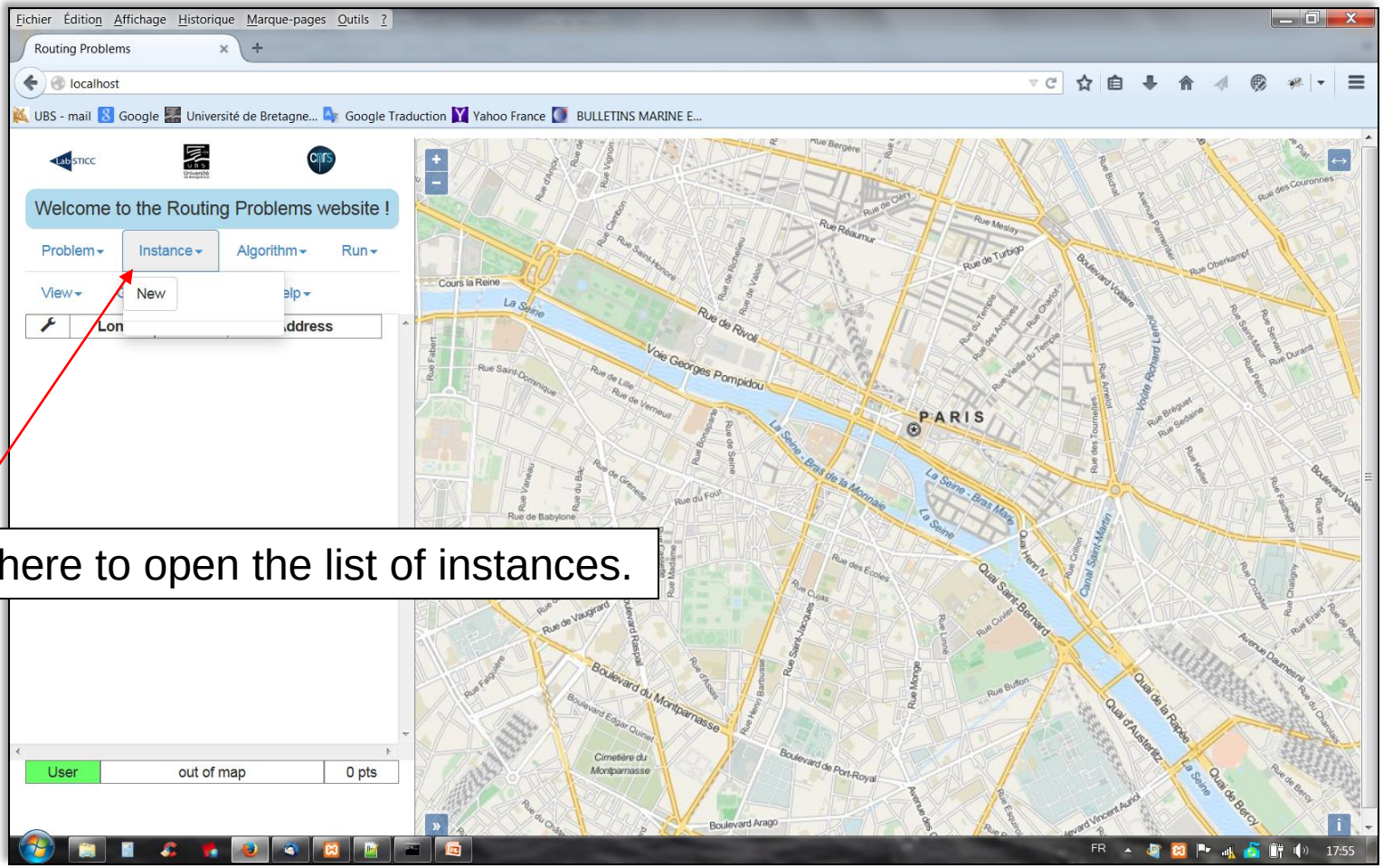


# Specifications: see all the problems.

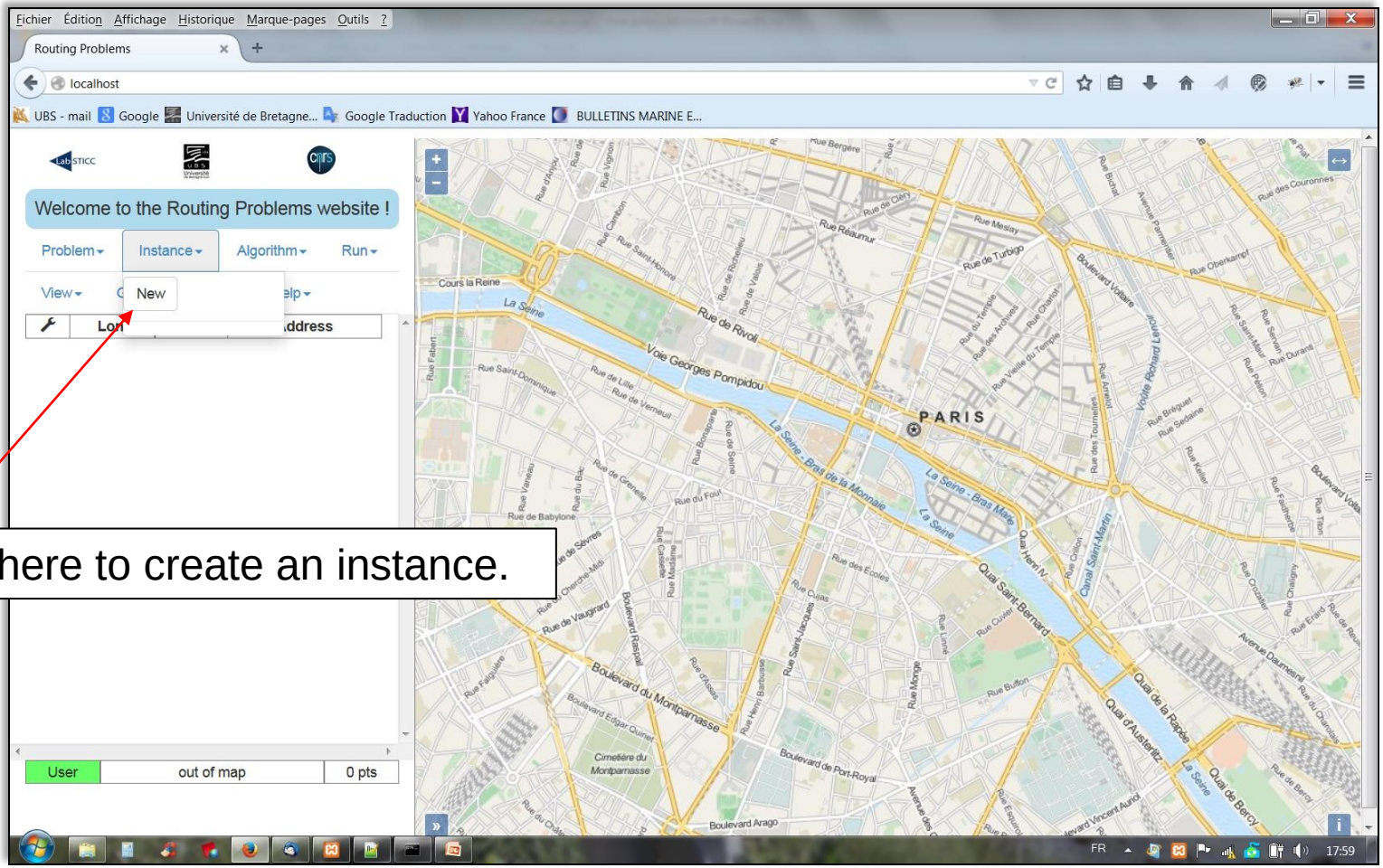



Click here to see all the problems.
A new problem called "line" has been added.

# Specifications: open the list of instances.

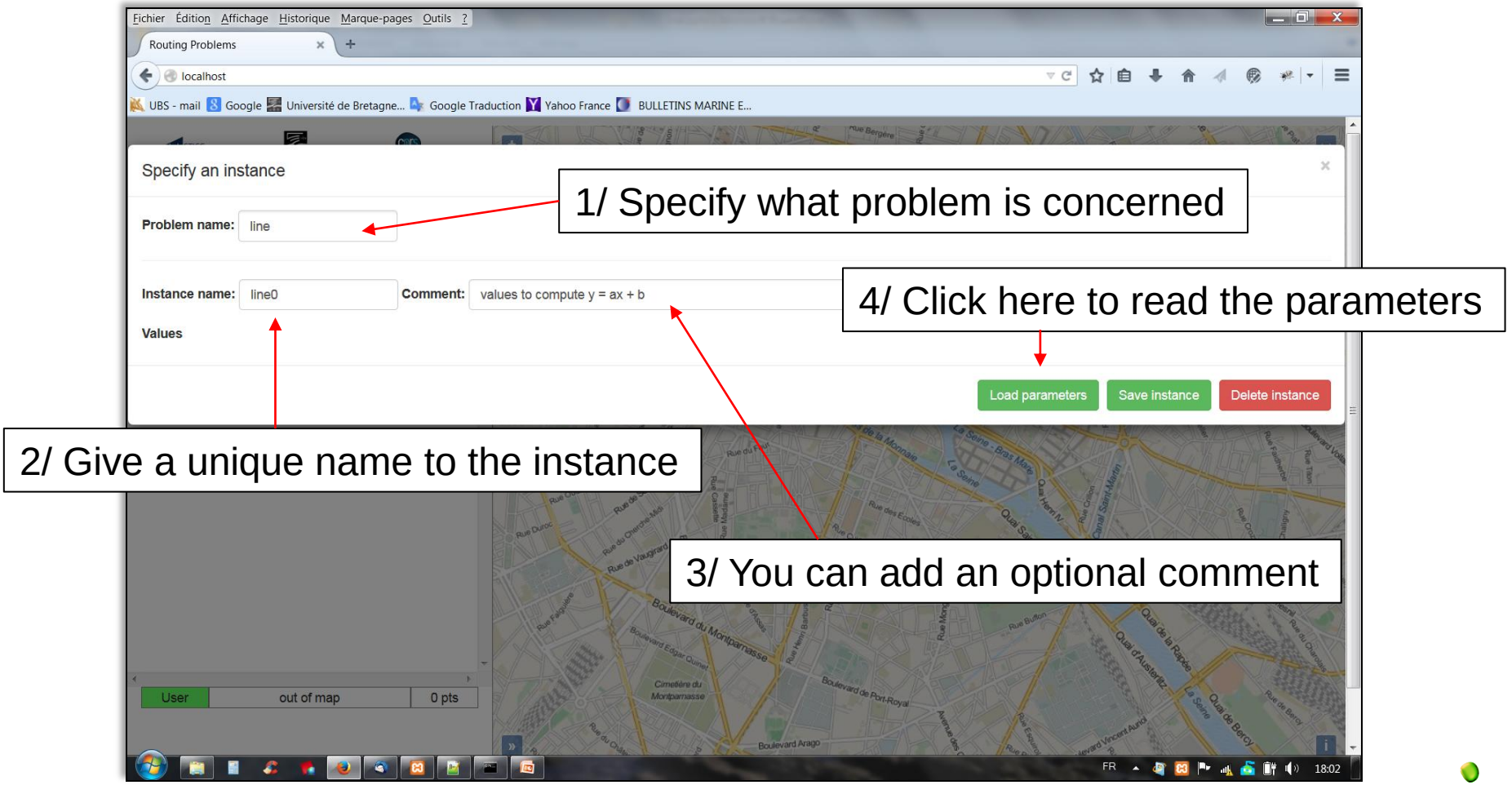


# Specifications: create an instance.



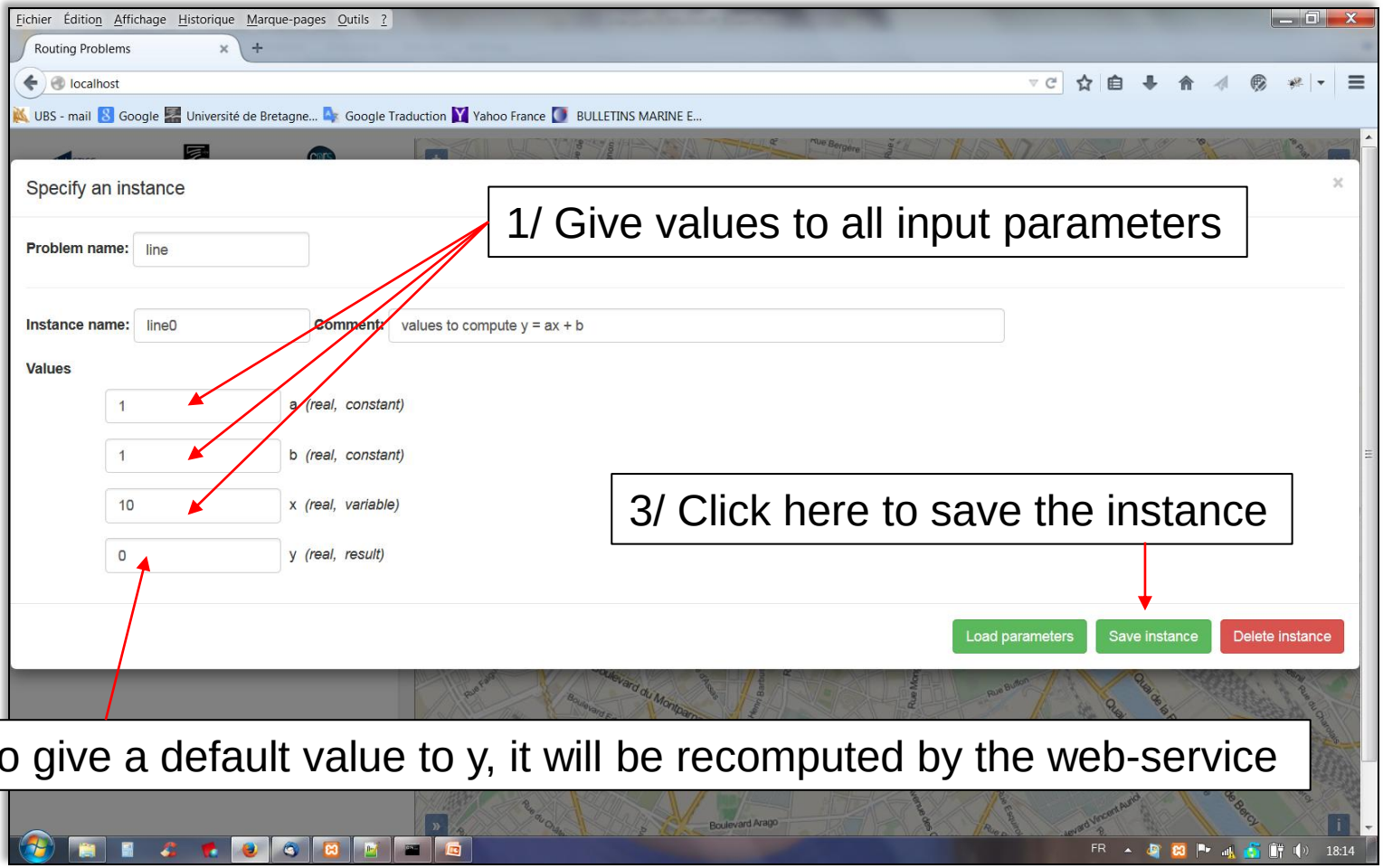

Click here to create an instance.

# Specifications: fill the form for the instance and give values to the parameters.

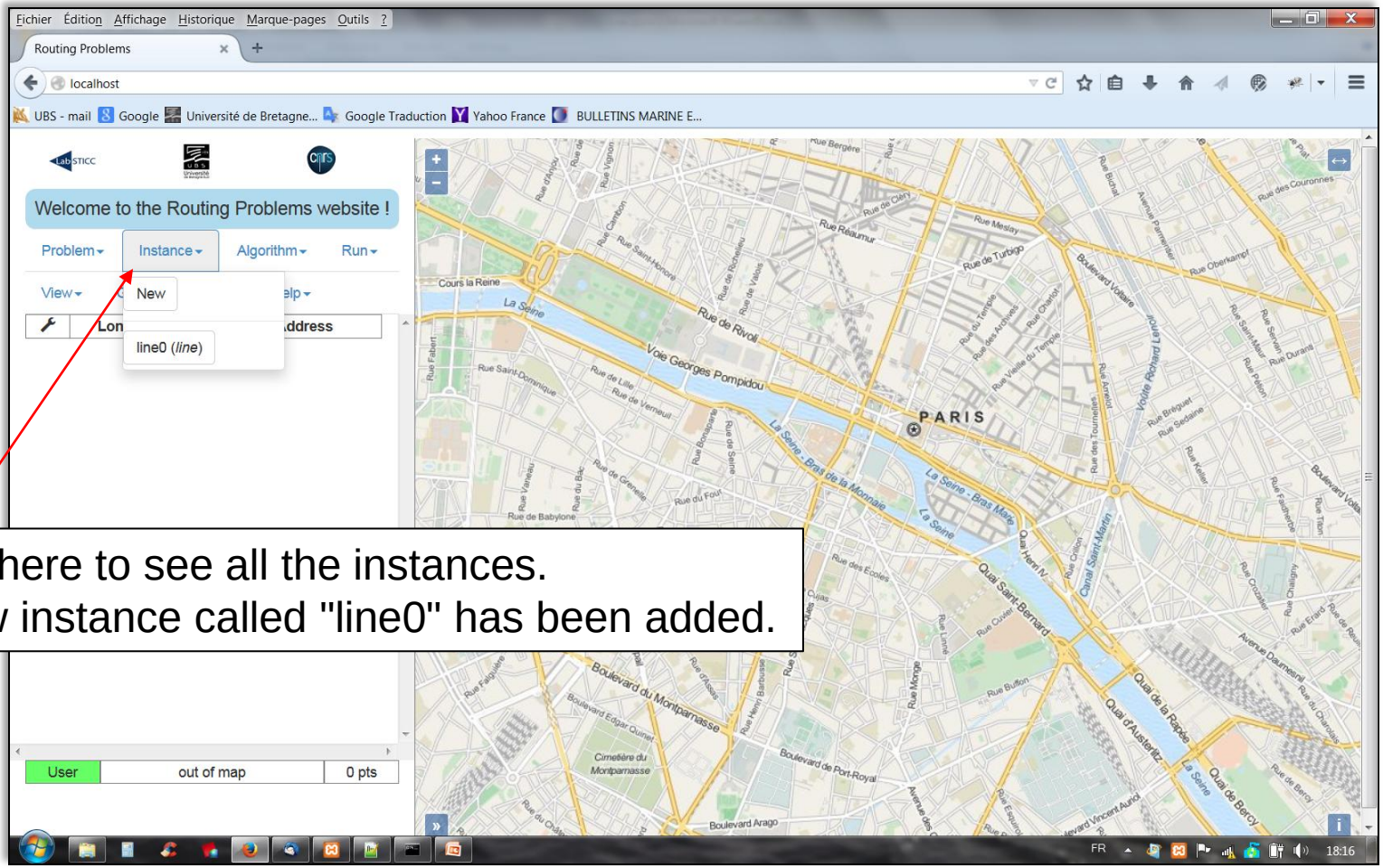



A window named « Specify an instance » opens.

# Specifications: fill the form for the instance and give values to the parameters.

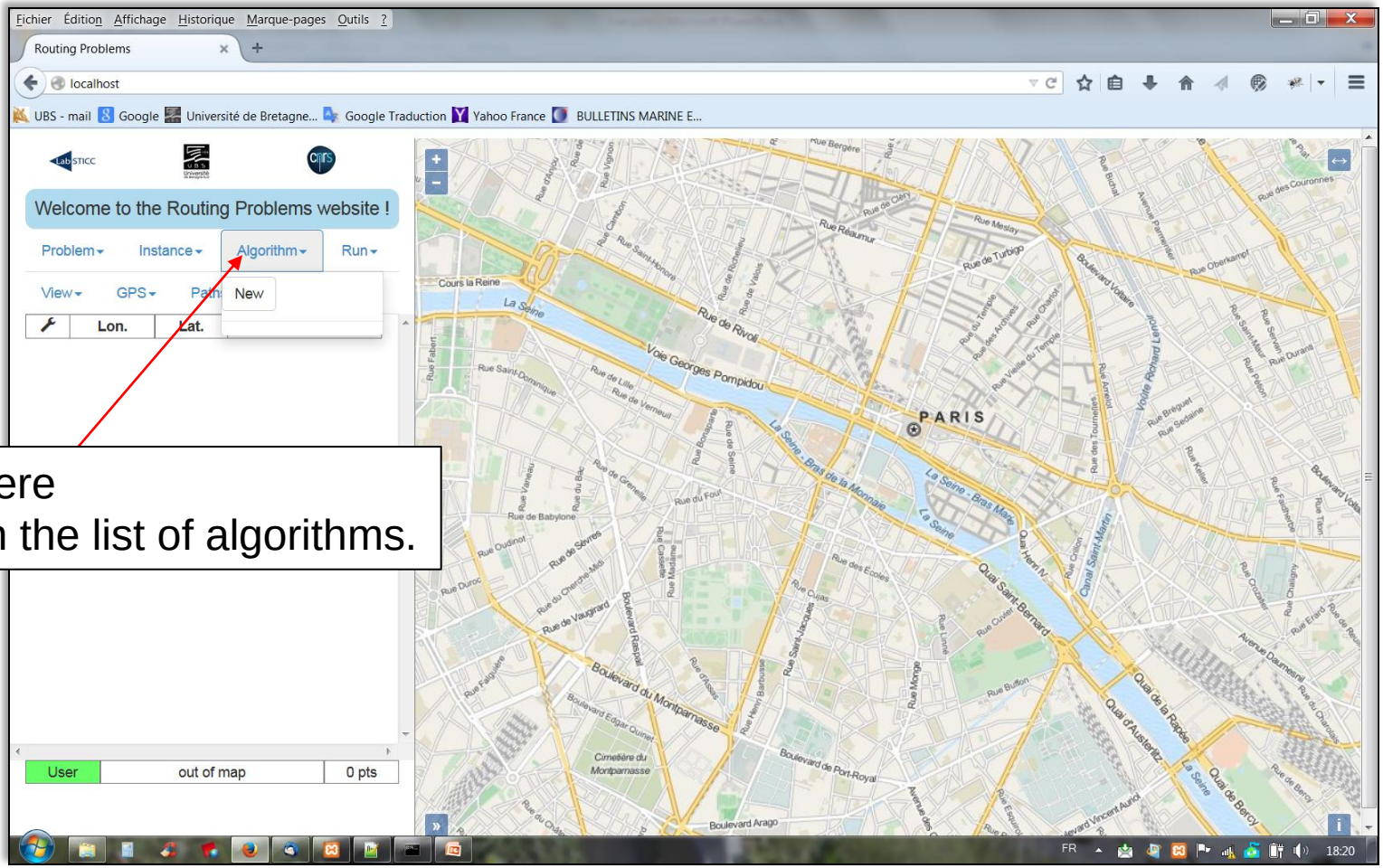


# Specifications: see all the instances.



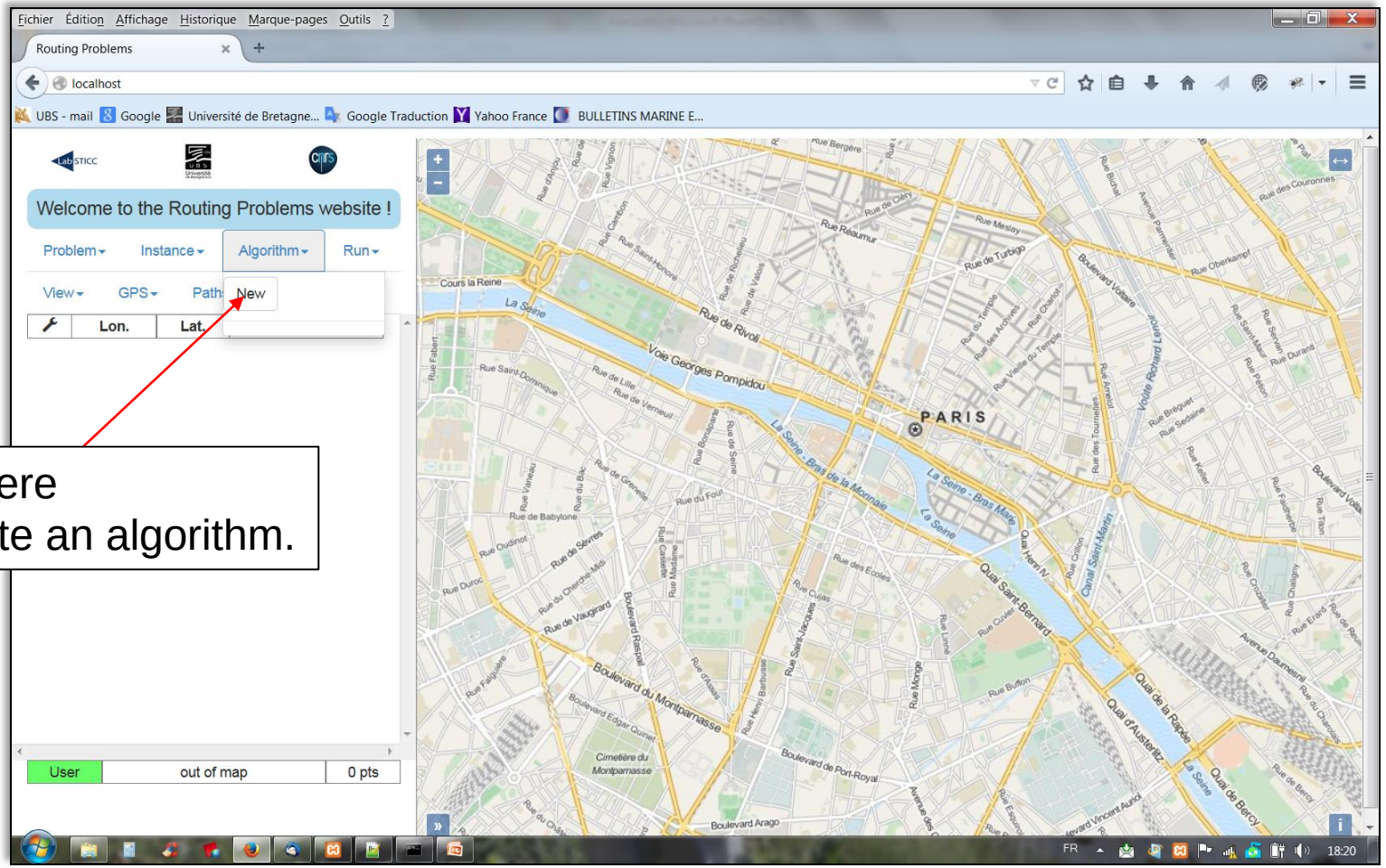

Click here to see all the instances.
A new instance called "line0" has been added.

# Specifications: open the list of algorithms.



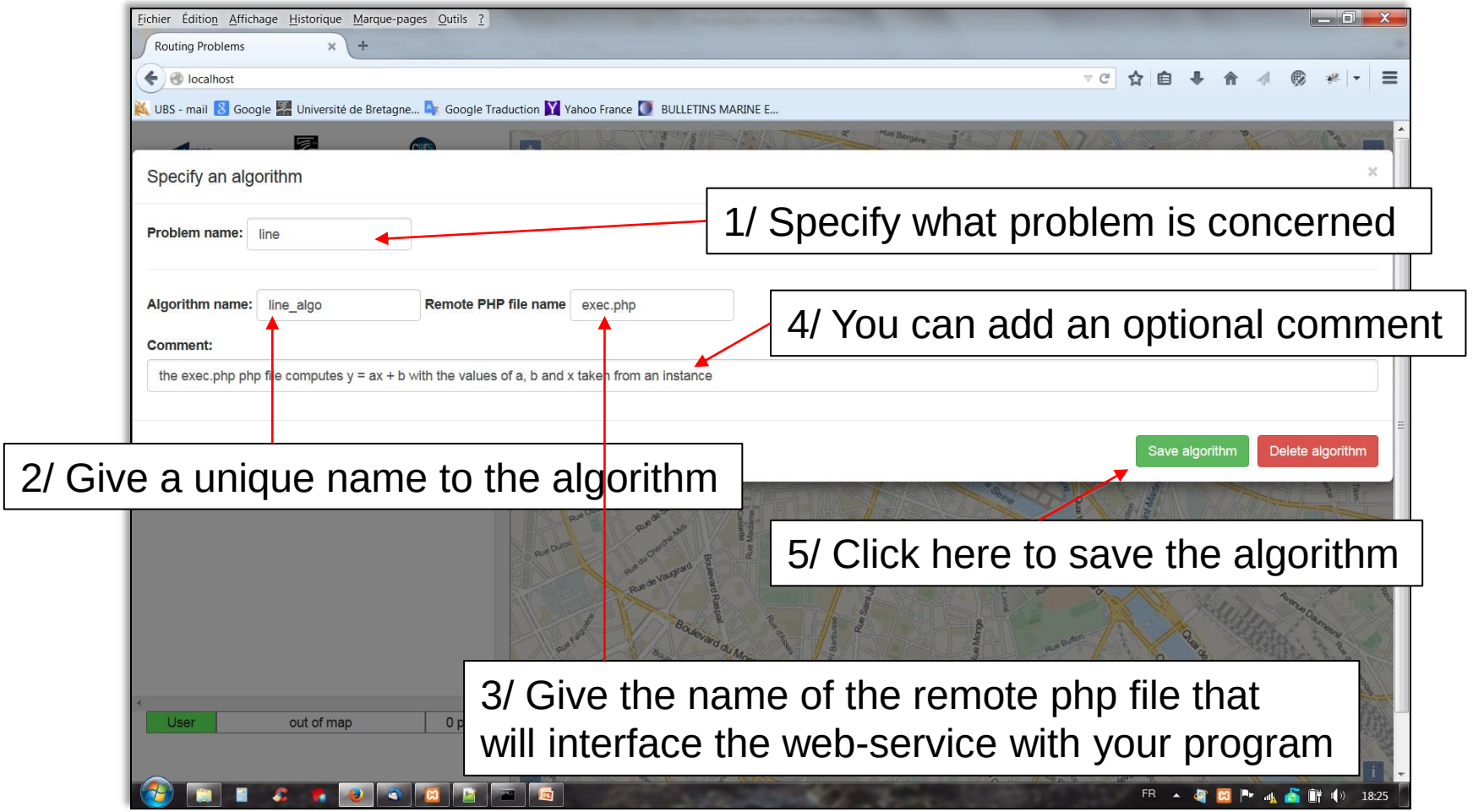

Click here
to open the list of algorithms.

# Specifications: create an algorithm.



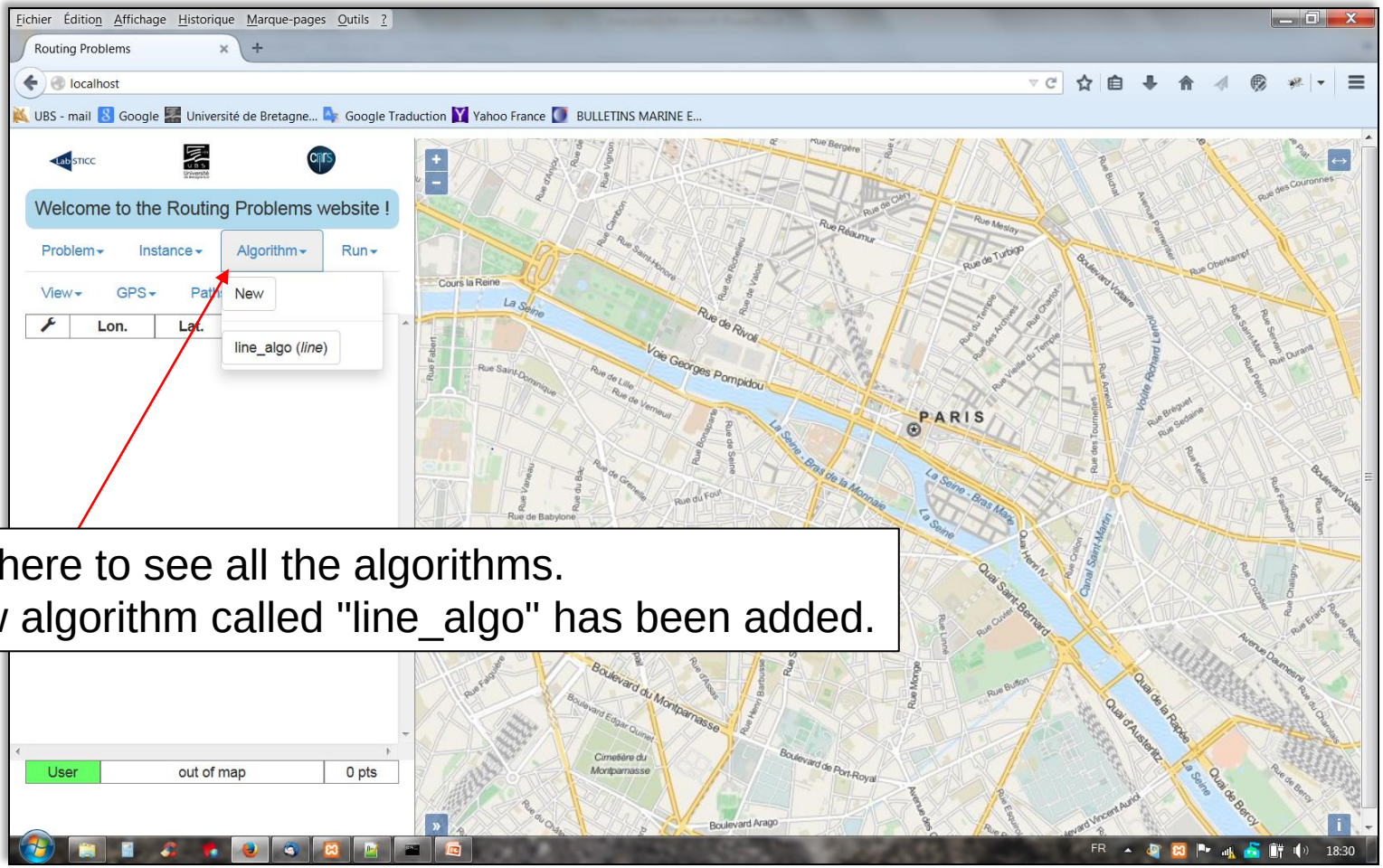

Click here
to create an algorithm.

# Specifications: create an algorithm.



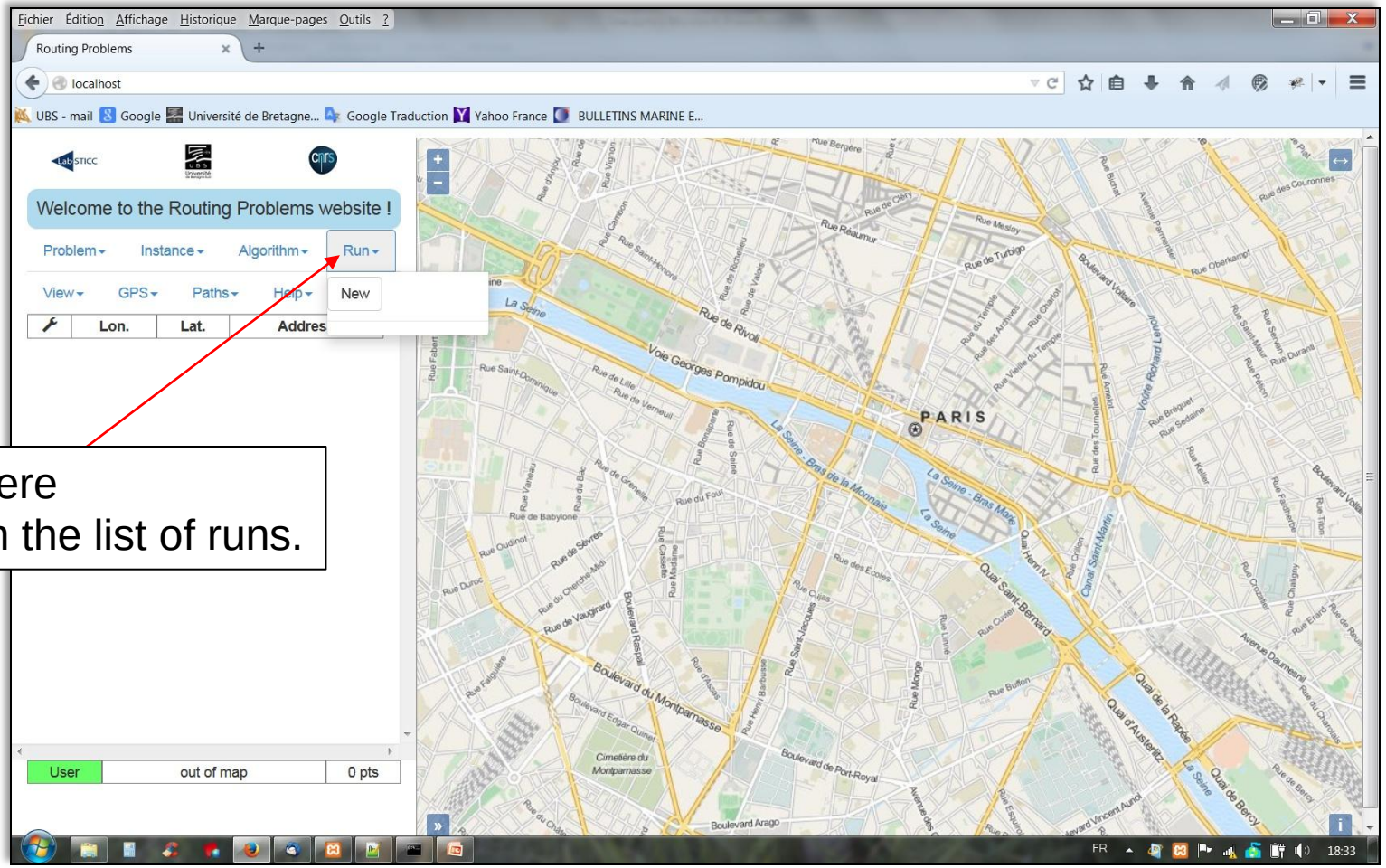

A window named « Specify an algorithm » opens.

# Specifications: see all the algorithms.

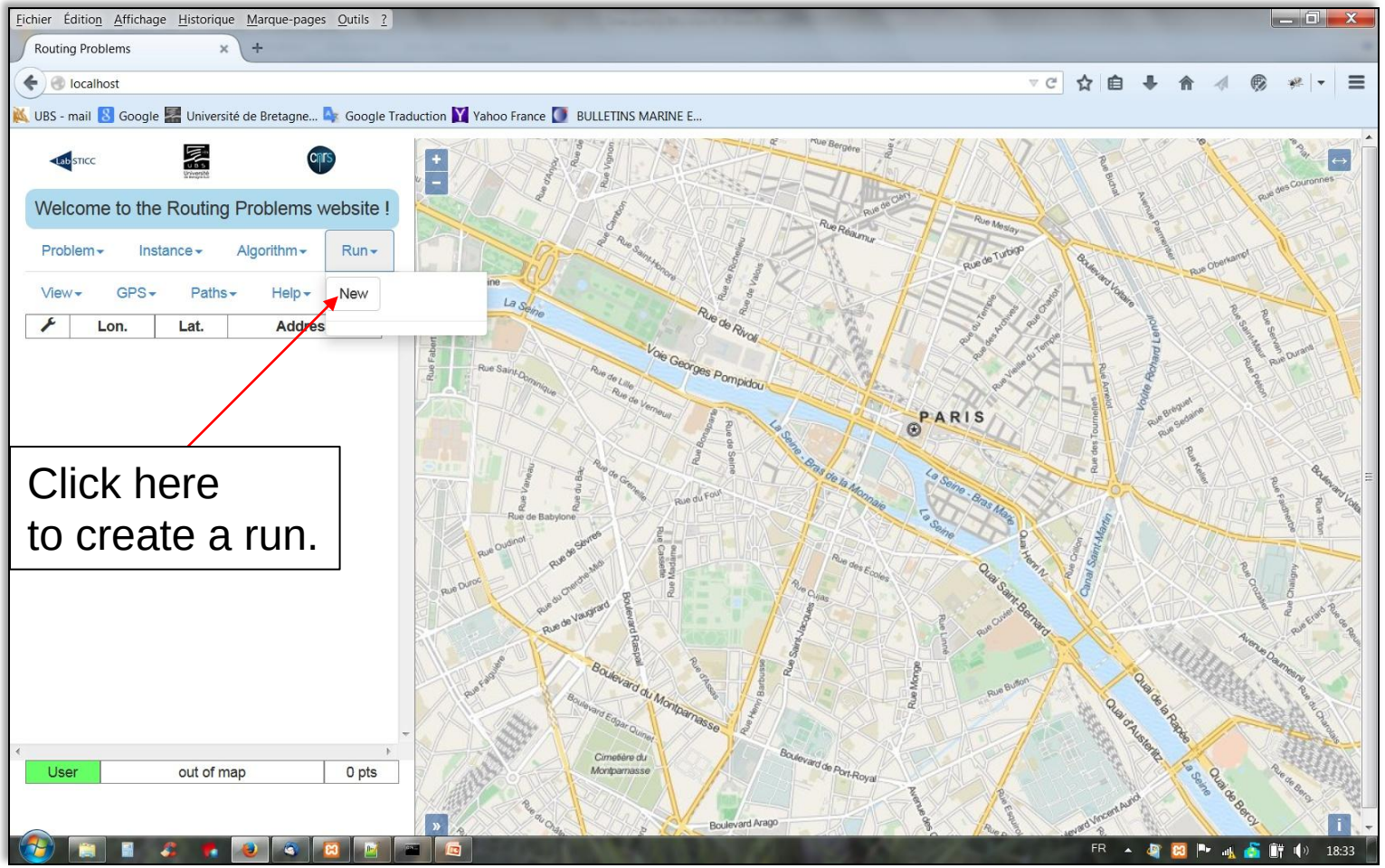



Click here to see all the algorithms.
A new algorithm called "line_algo" has been added.

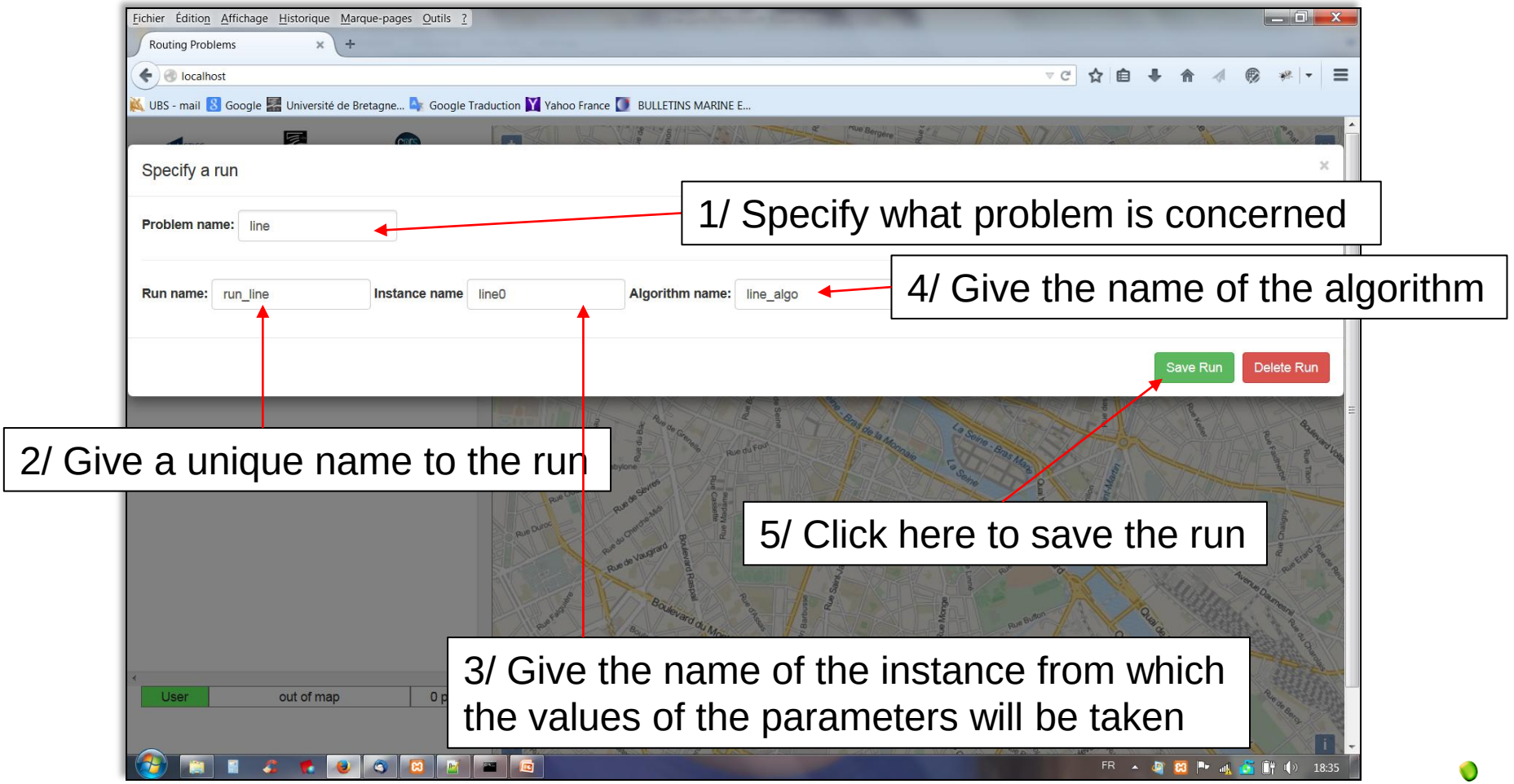
# Specifications: open the list of runs.



# Specifications: create a run.

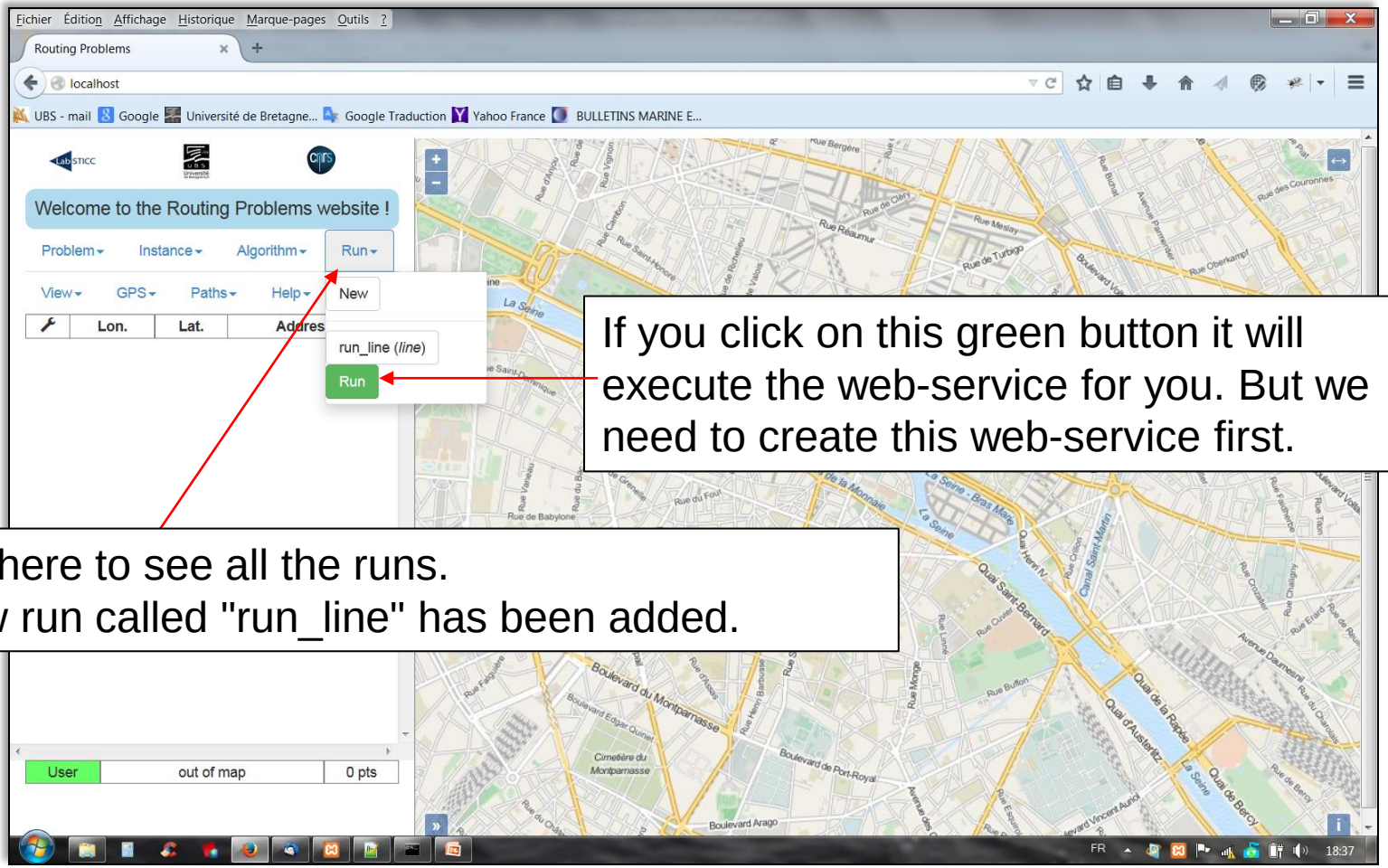


# Specifications: create a run.

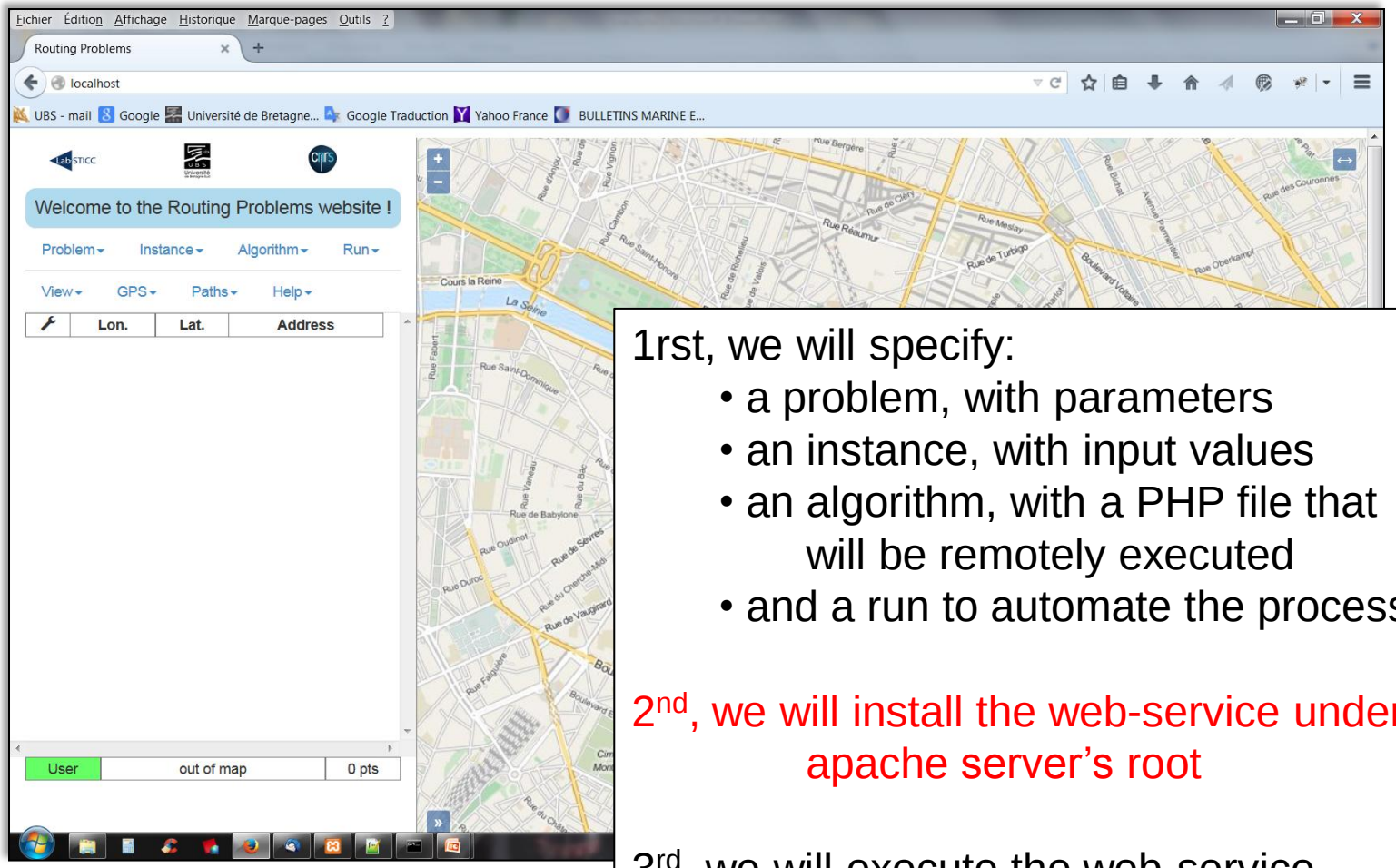



A window named « Specify a run » opens.

# Specifications: see all the runs.



# Installation of the web-service



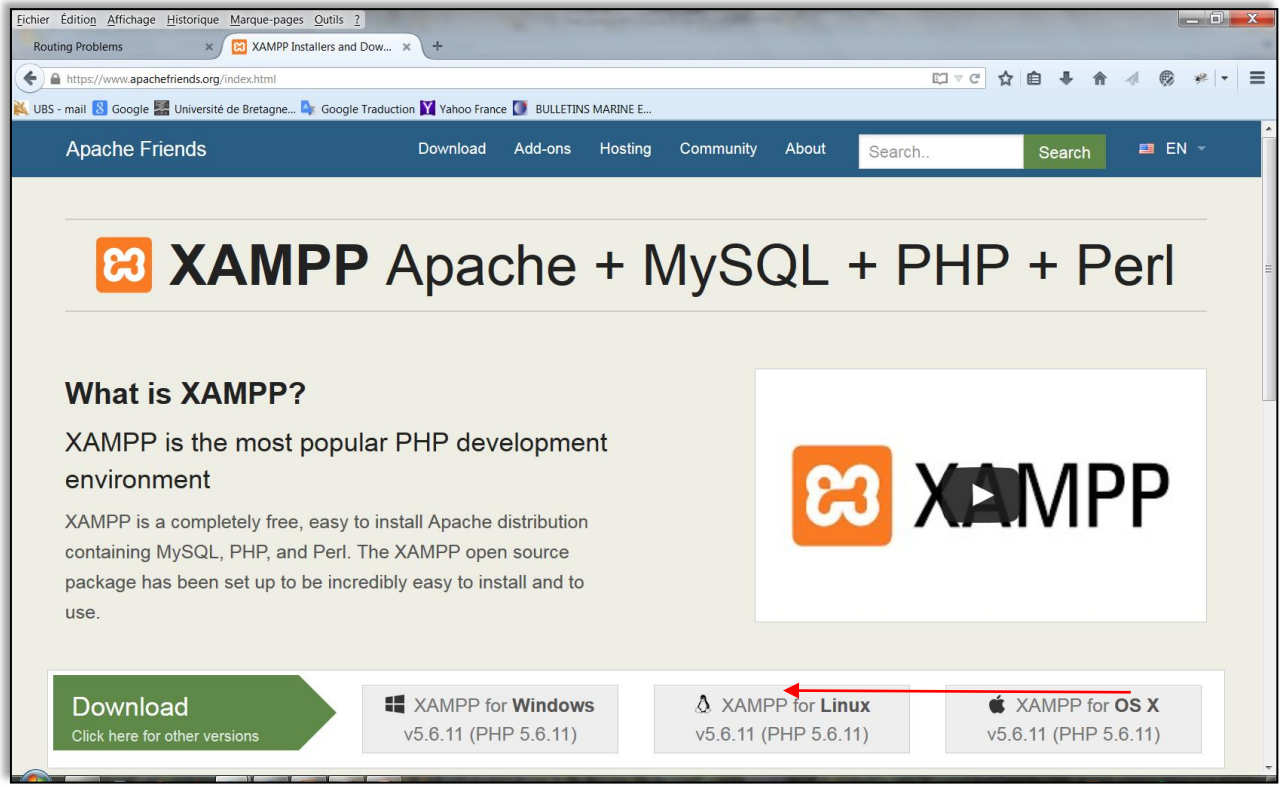

1rst, we will specify:

- a problem, with parameters
- an instance, with input values
- an algorithm, with a PHP file that will be remotely executed
- and a run to automate the process

2nd, we will install the web-service under the apache server's root

3rd, we will execute the web-service

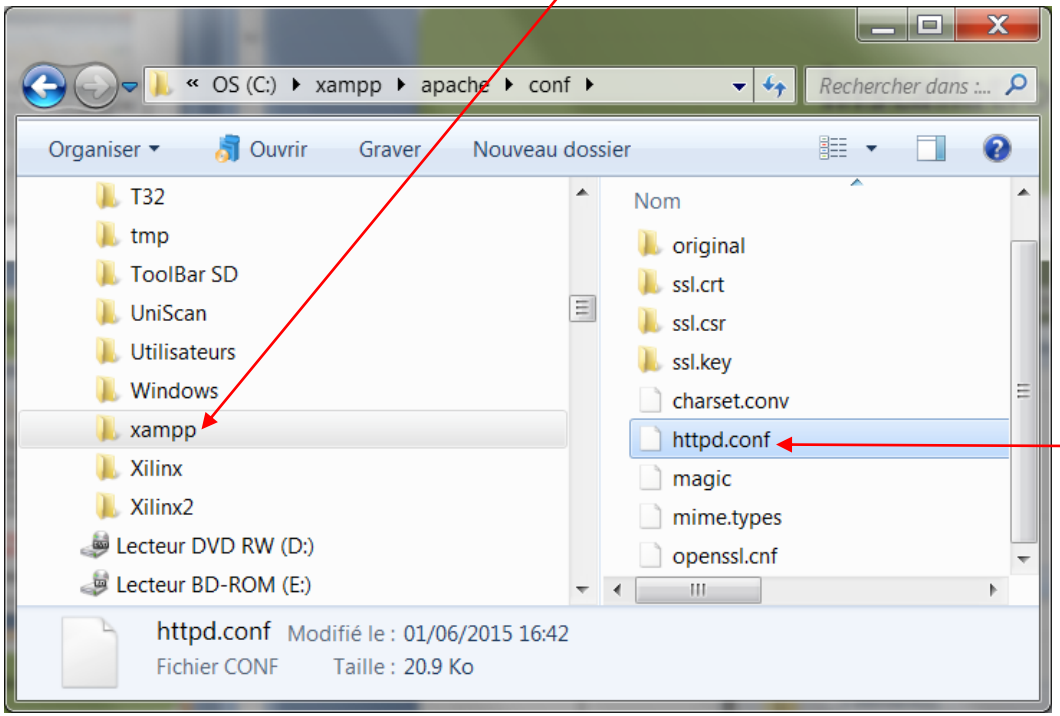
# Installation of the web-service




If you need to install XAMPP
go to https://www.apachefriends.org/index.html
and follow the instructions.

# Installation of the web-service

The Apache server we have selected is XAMPP.
It should be installed on the C: drive
Under the *xampp* directory.

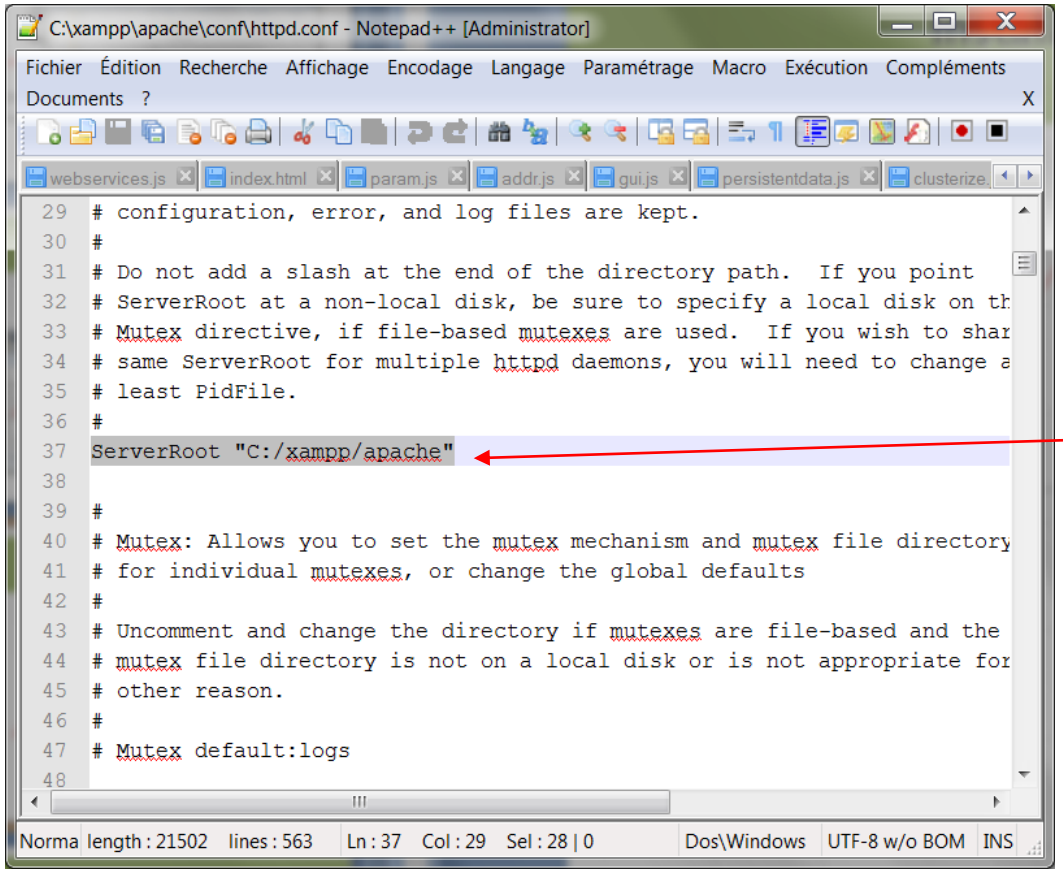



The configuration file is *httpd.conf*

You will modify it to locate your web-services.

Open it with an editor.

# Installation of the web-service



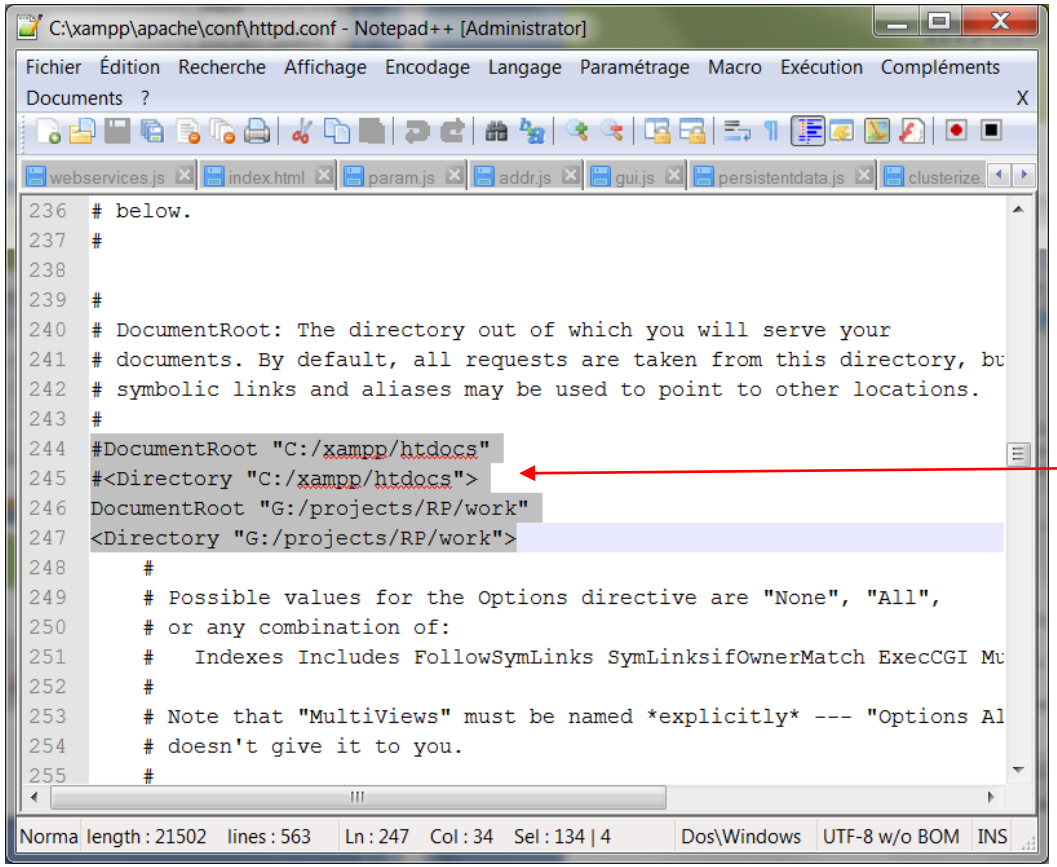

Look at the *ServerRoot* directive.

It should point to the location of the XAMPP installation of the apache server.

You should not modify it once it has been specified, unless you want to reinstall XAMPP somewhere else on your disk.

XAMPP has been installed somewhere, *ServerRoot* contains this location.

# Installation of the web-service



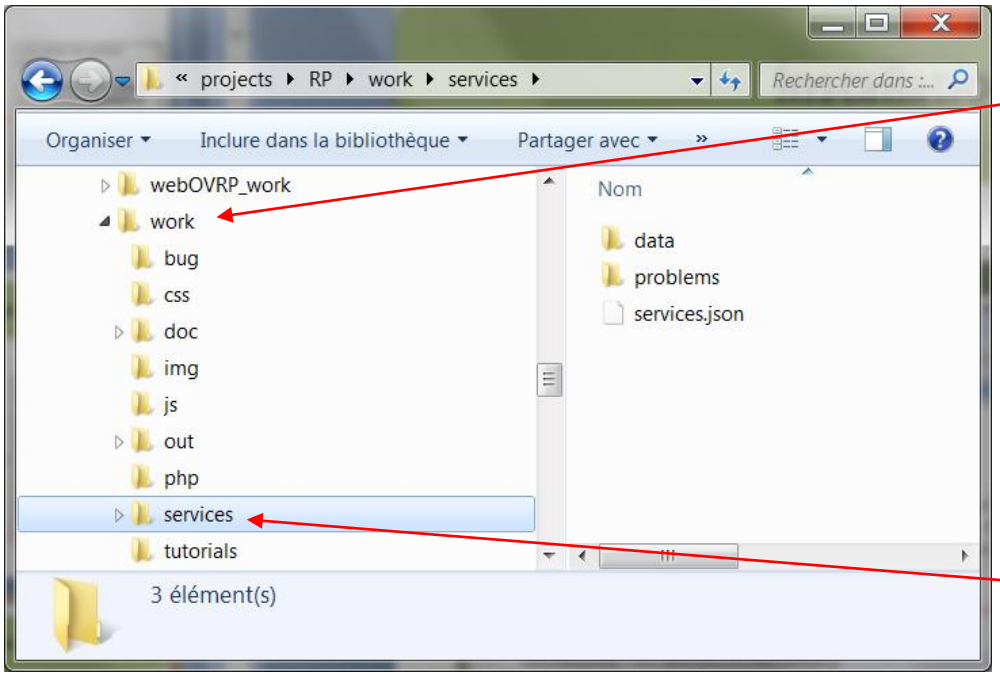

Look at the *DocumentRoot* directive.

It should point to the location of the private directory of the apache server.

You should not modify it once it has been specified, unless you want to move this directory somewhere else on your disk.

Apache works with private data, *DocumentRoot* contains this location.

# Installation of the web-service




Open a file explorer and go to the location of the web-services.

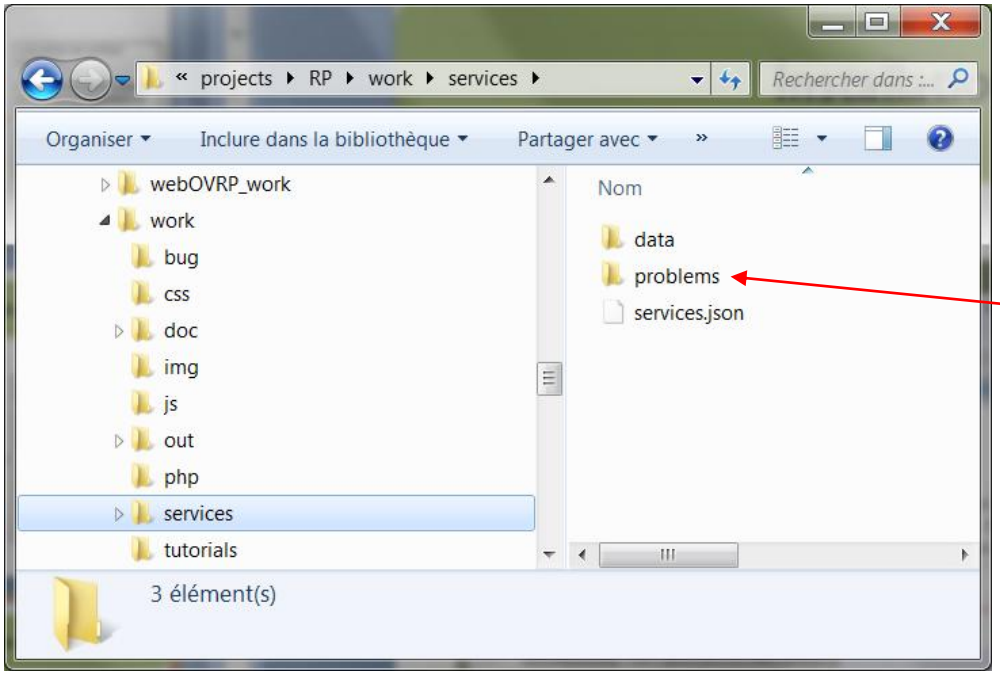
This location is the value of the *DocumentRoot* directive.

You may see several directories.

Among them, the *services* directory is the one where are stored all the web-services.

If it does not exist, create it.

# Installation of the web-service

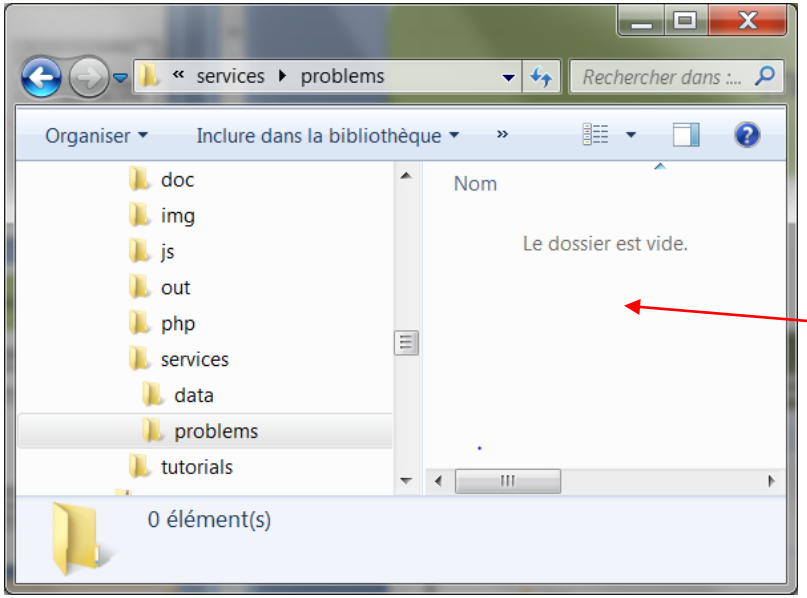



Under *services* there is another directory named *problems*.

This is where all the web-services (called problems in OR) are stored.

Open it.

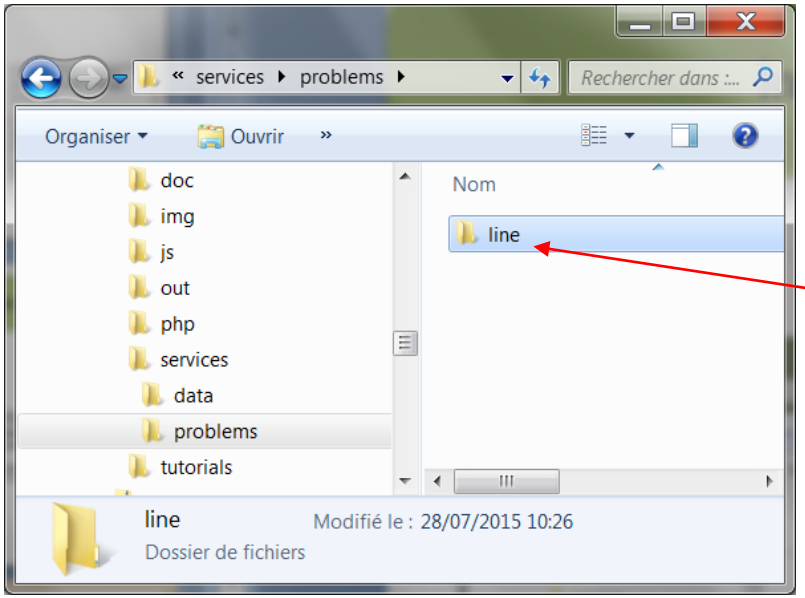
# Installation of the web-service




This directory is empty because you have currently no web-service.

We'll add the *line* web-service.

# Installation of the web-service



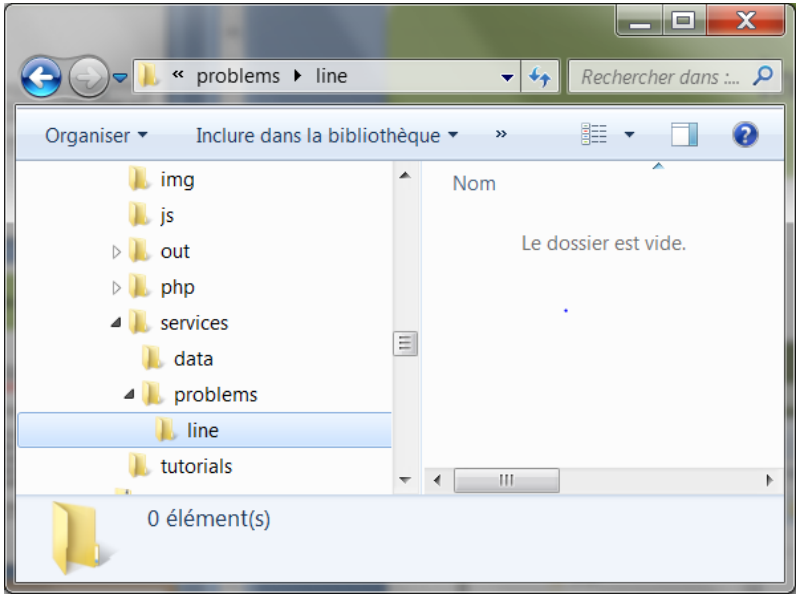

Rule1 : each problem is identified by a unique name.

In our case it is *line*.

To represent this problem on the web-server's side, you must create a directory with the same name than the problem.

Create the *line* directory here.

# Installation of the web-service




Of course this new directory is empty.

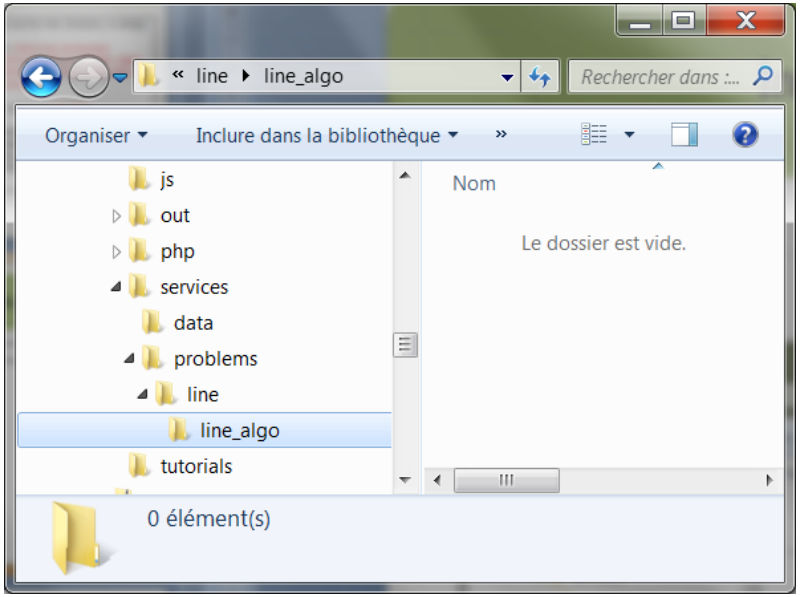
Rule 2: there may be several algorithms to solve a problem. Each algorithm has a unique name, for a given problem.

In our case it is *line_algo*.

To represent this algorithm on the web-server's side, you must create a directory with the same name than the algorithm.

Create the *line_algo* directory here.

# Installation of the web-service



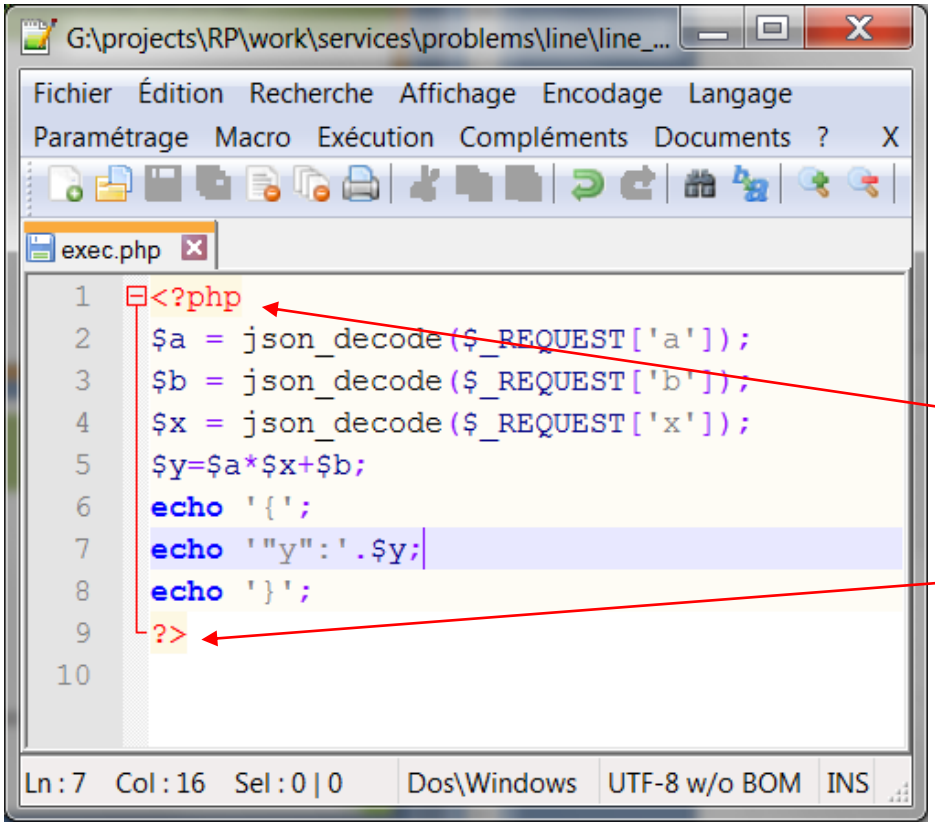

Of course this new directory is empty.

Do you remember the name we have choosen for the PHP file while creating the algorithm ?

It was *exec.php*

So, use an editor and create a php file with name *exec.php* here.

# Installation of the web-service



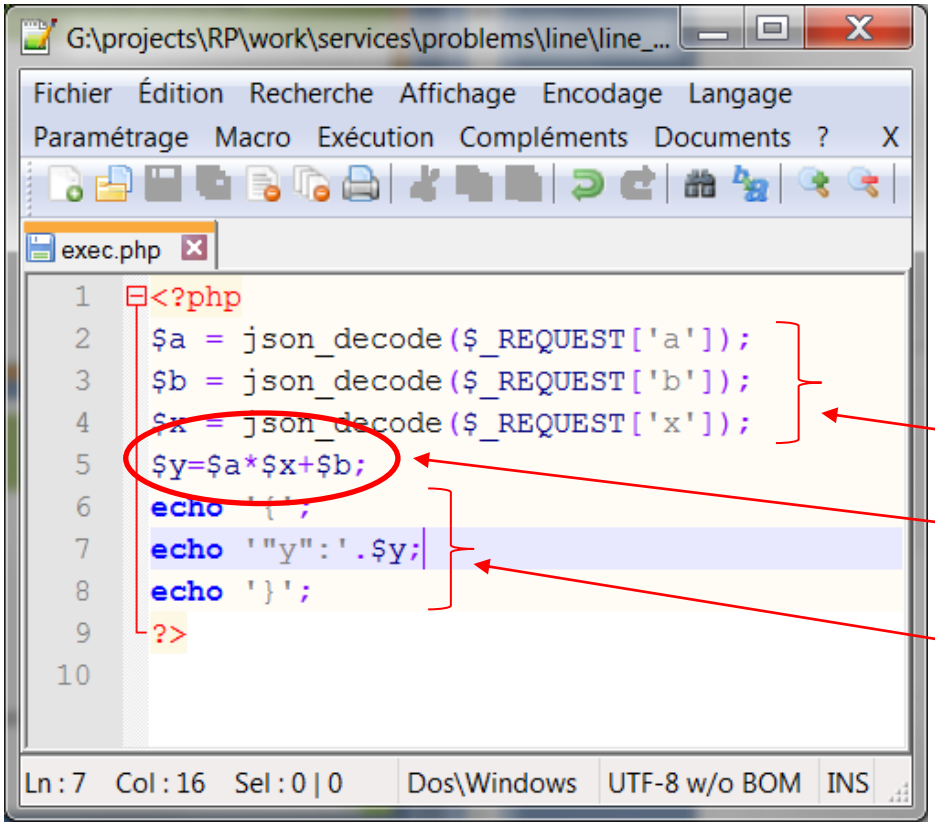

Learning PHP is out of the scope of this tutorial, but we think that it is simple enough to be understood very quickly.

A php file always has

a header

and a footer.

# Installation of the web-service



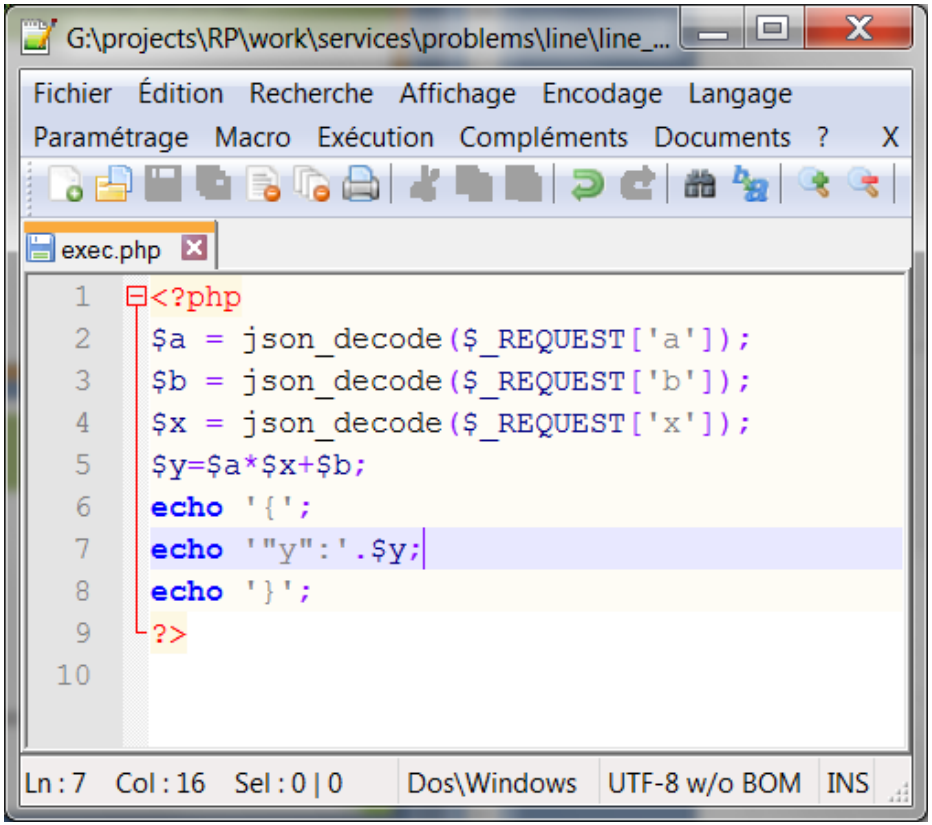

A web-service does 3 things:

1/ read values of parameters

2/ do some processing

3/ return values of results

# Installation of the web-service




The GUI of the RP web-site has been developped with Javascript. All the transfered values are JSON coded and identified by their corresponding parameter's name.

To get the value of the *a* parameter, one has to write:

```
$_REQUEST['a']
```

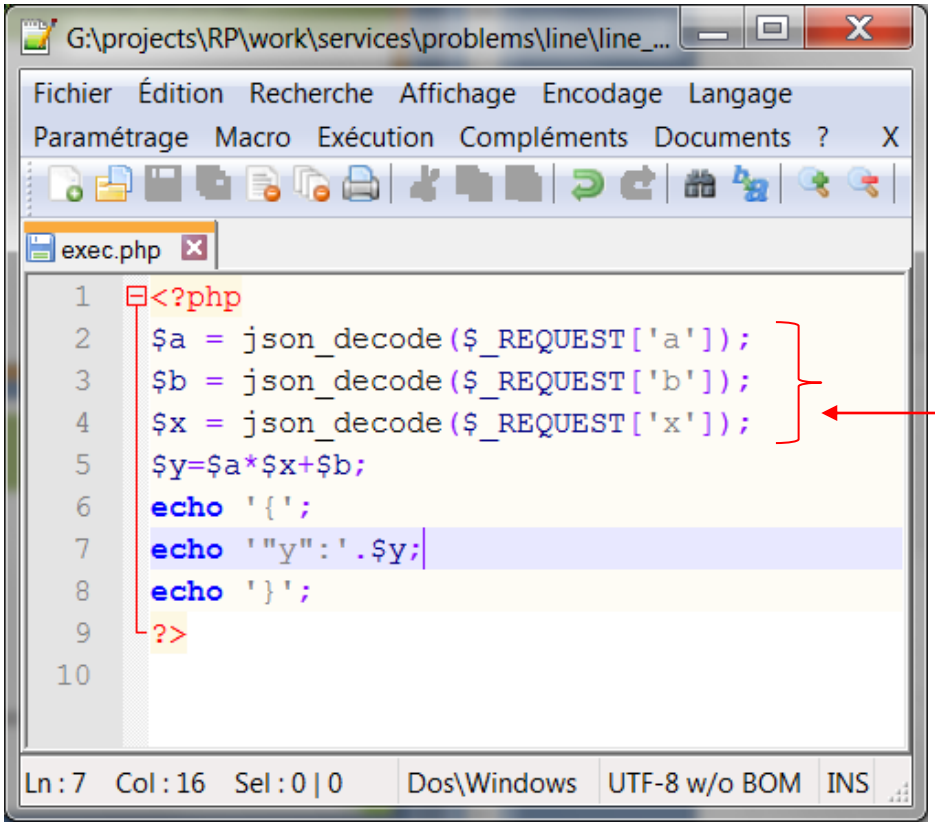
to decode it:

```
json_decode($_REQUEST['a'])
```

to copy the value into a variable:

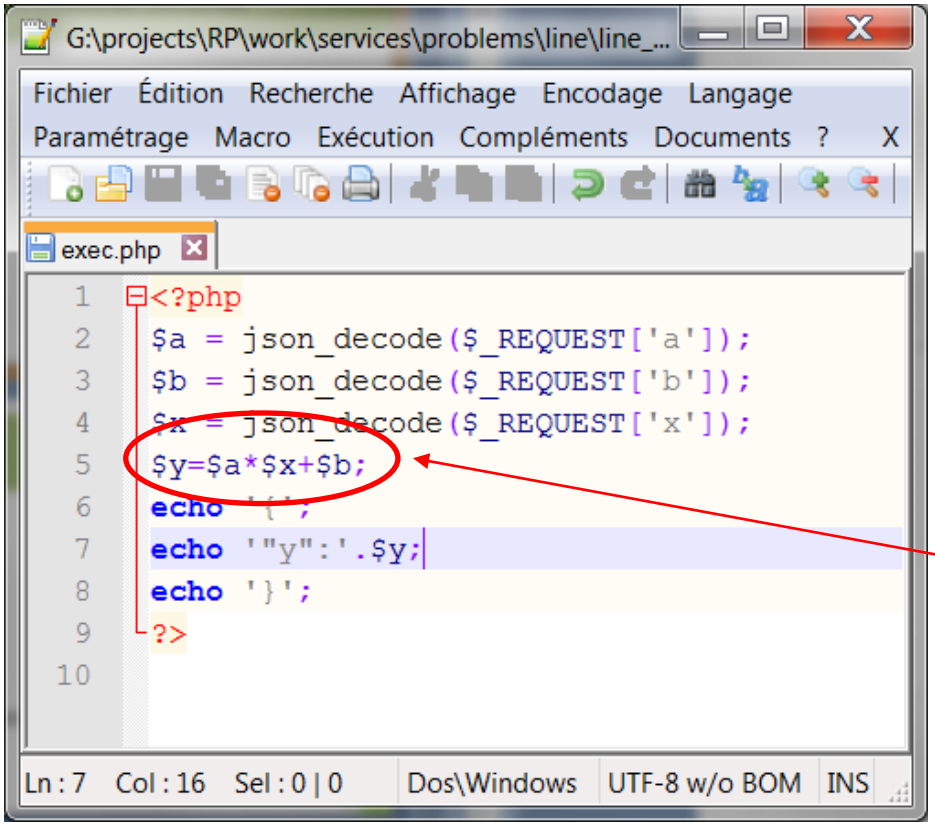
```
$a = ...
```

# Installation of the web-service




As you can see, the start of the php file reads the values of the *a*, *b*, and *x* parameters and copy them into the variables *$a*, *$b* and *$x*.

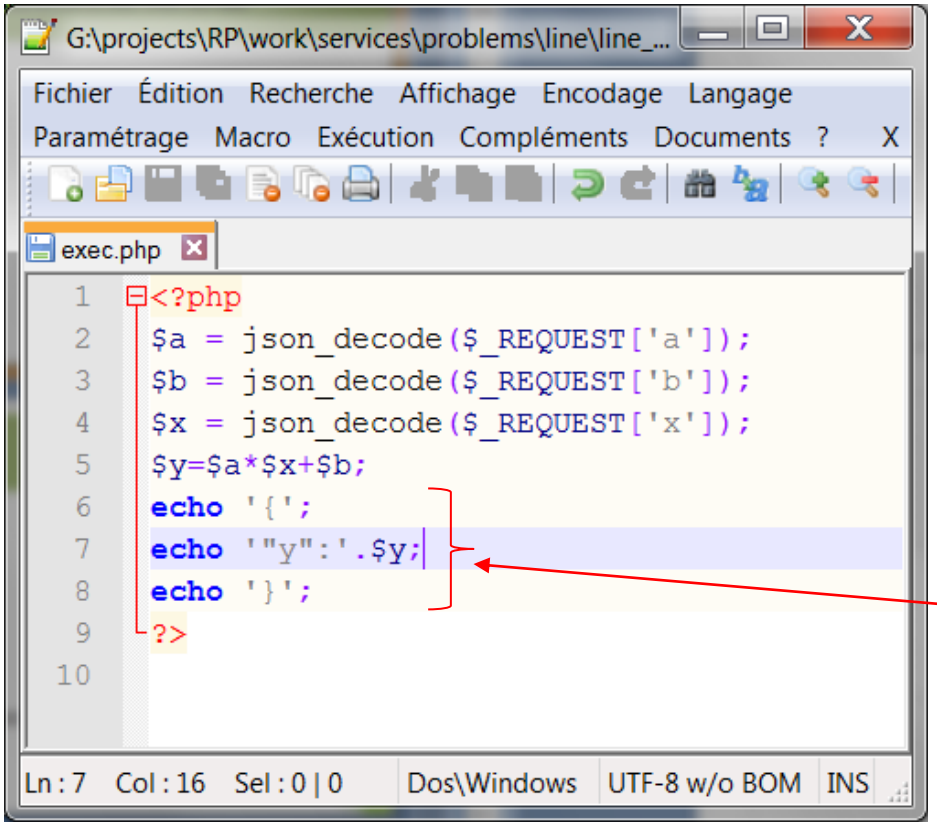
# Installation of the web-service




The processing part is here the simplest.

We compute into the *$y* variable the value *$a*$x+$b*, which is exactely what we wanted to do.

# Installation of the web-service




The result part may not seem obvious, as long as you do not know what JSON is.

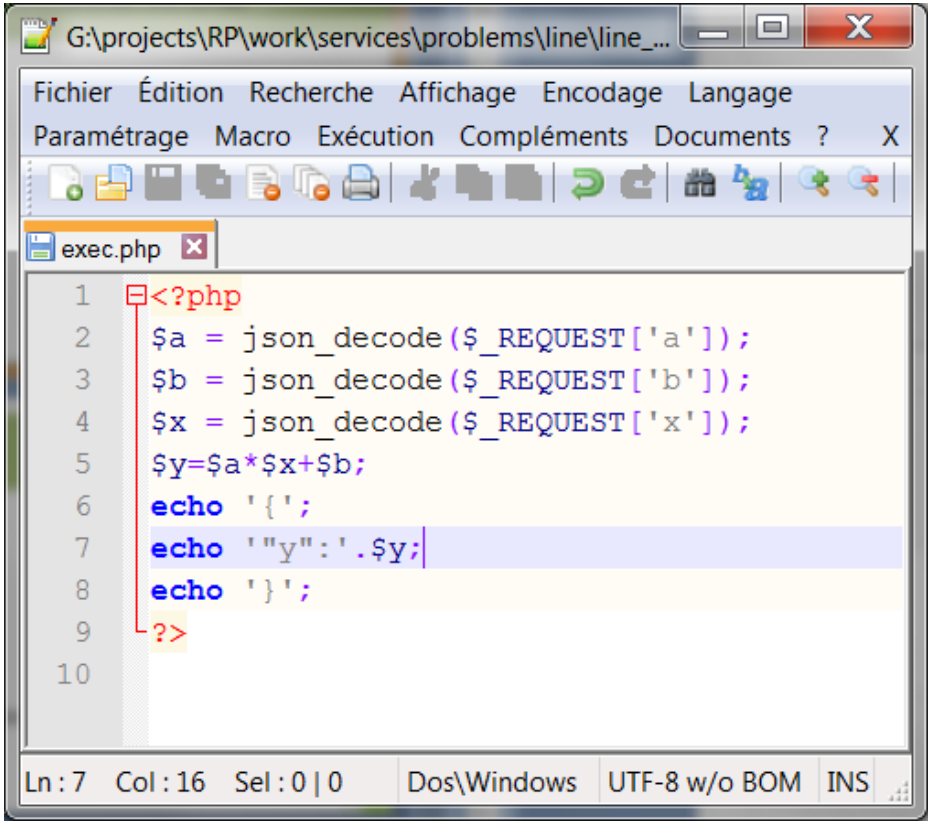
But, believe us, it is very easy to understand.

Here, we send back a string that is JSON coded. This string is exactely:

```
{"y":0}
```

Of course y = 0, because we assume that $a, $b and $x were all equal to 0 in that case.

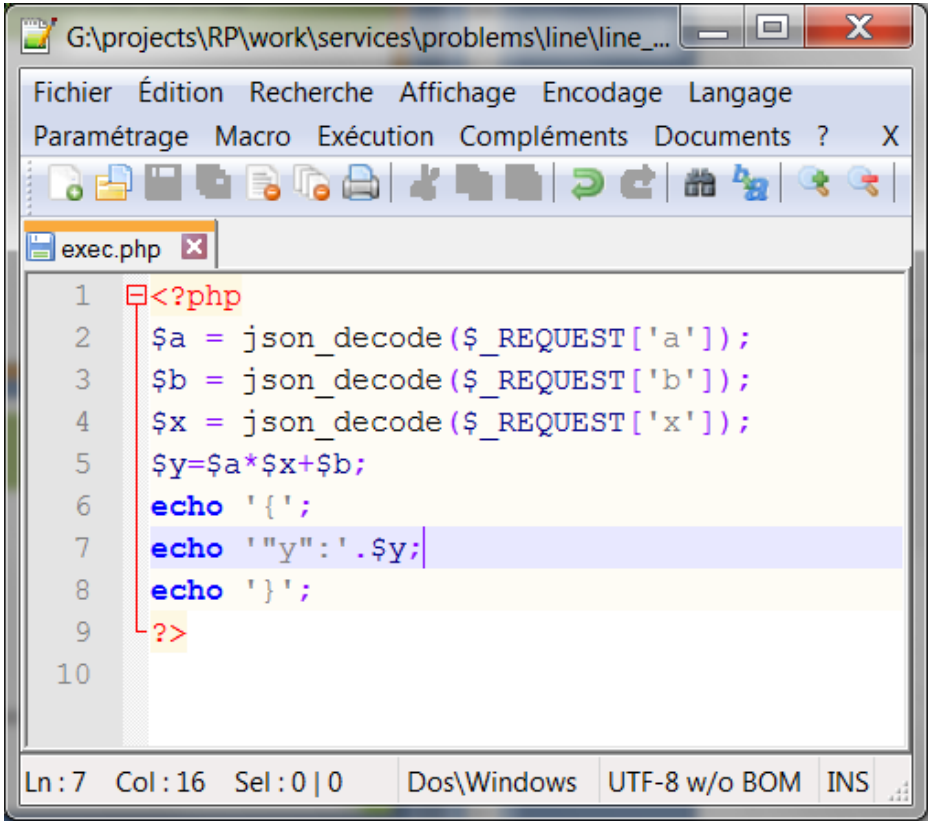
# Installation of the web-service




Rq: if we had several values to send back, we would have sent a string like this one:

```
{"y1":0, "y2":1}
```

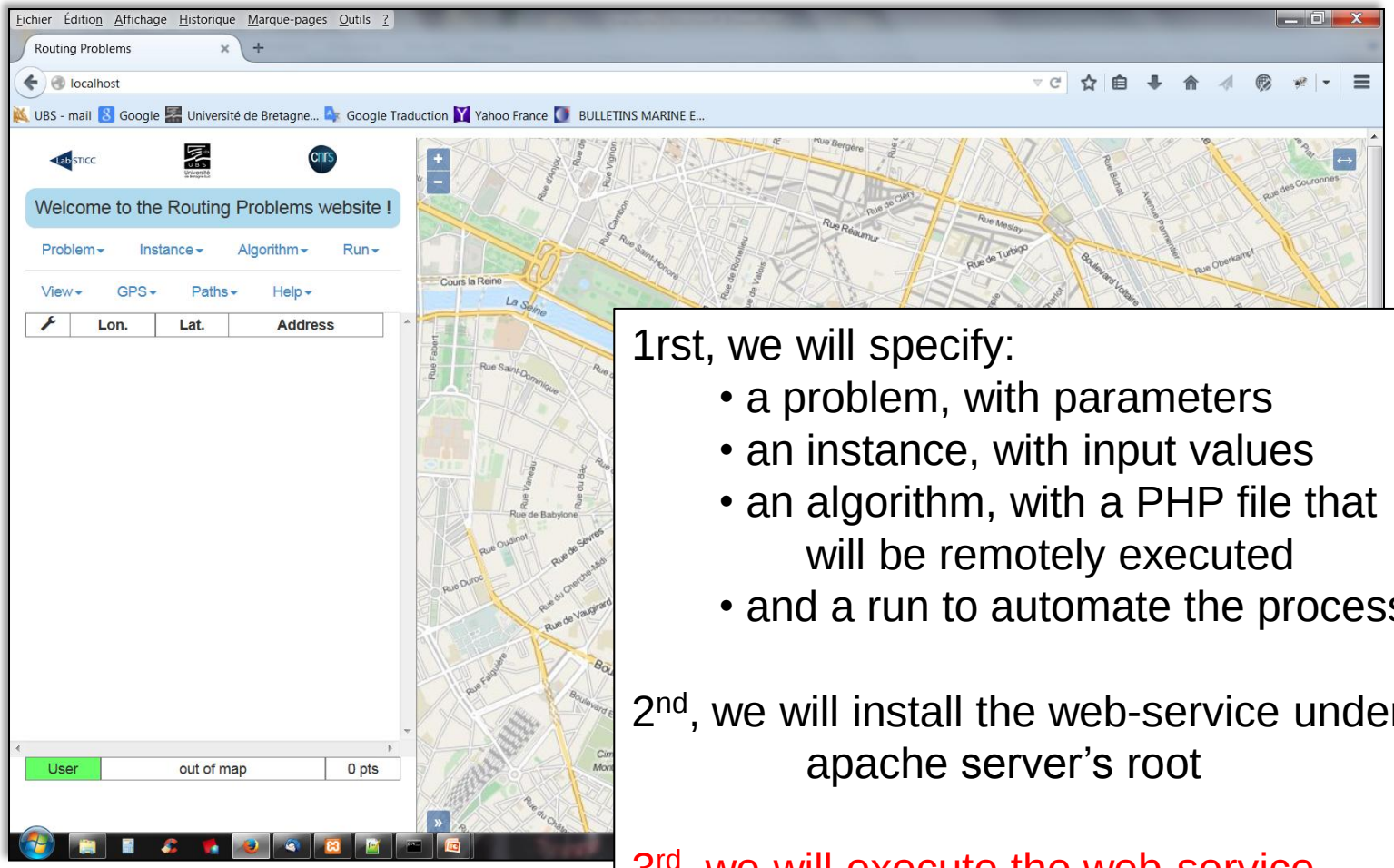
# Installation of the web-service




This is all for the *line* web-service.

Save the file and exit.

# Execution of the web-service



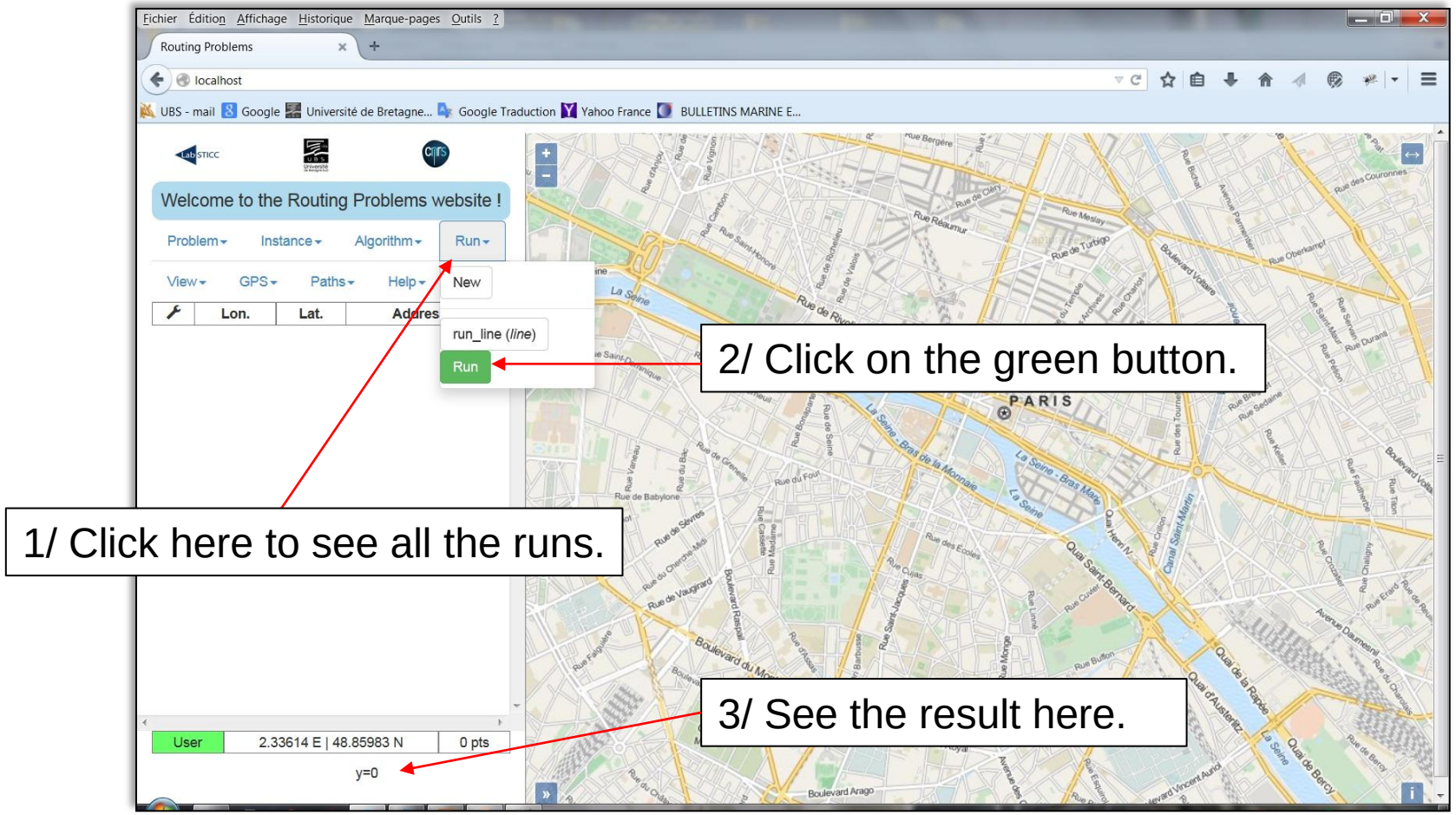

1rst, we will specify:

- a problem, with parameters
- an instance, with input values
- an algorithm, with a PHP file that will be remotely executed
- and a run to automate the process

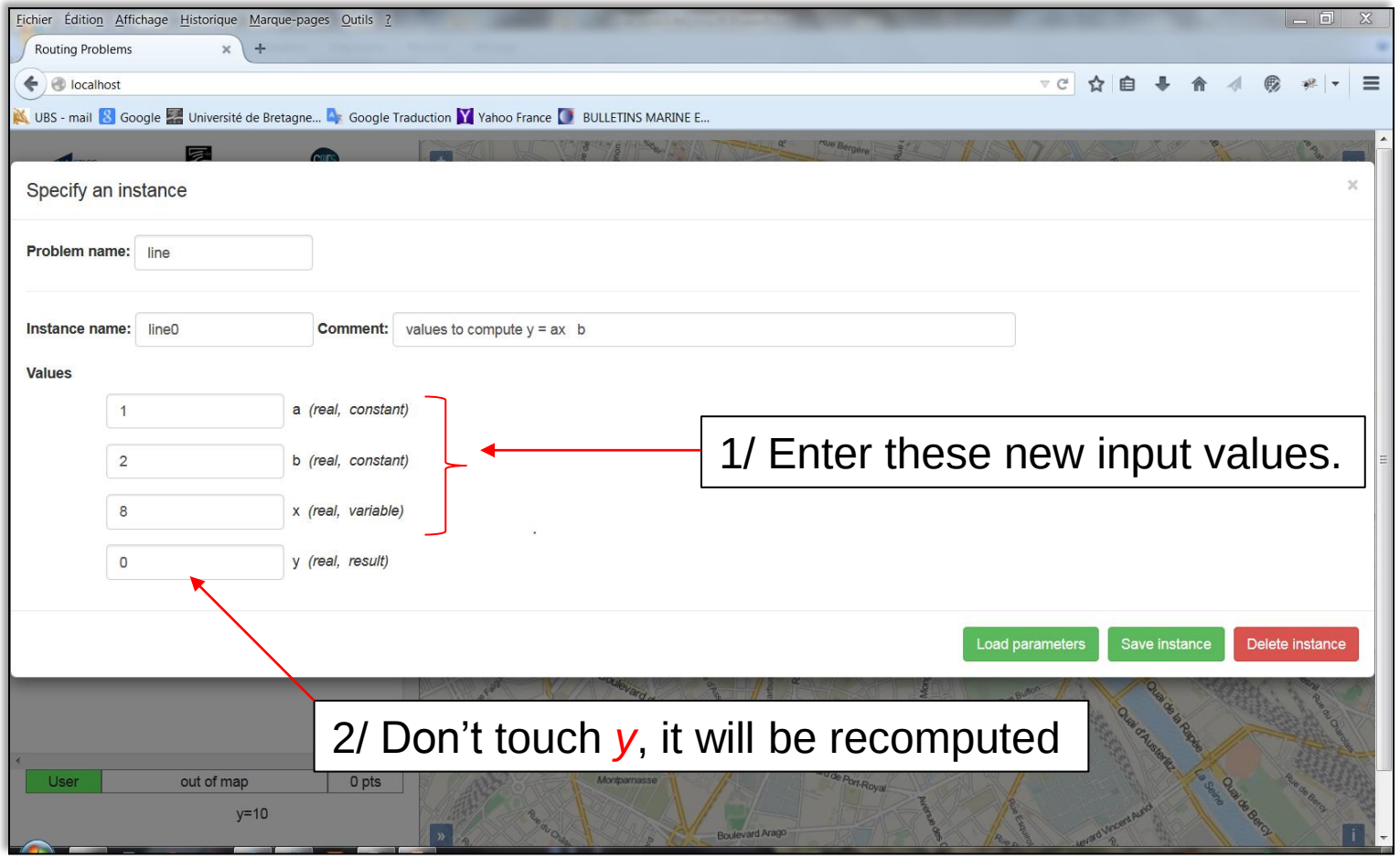
2nd, we will install the web-service under the apache server's root

3rd, we will execute the web-service

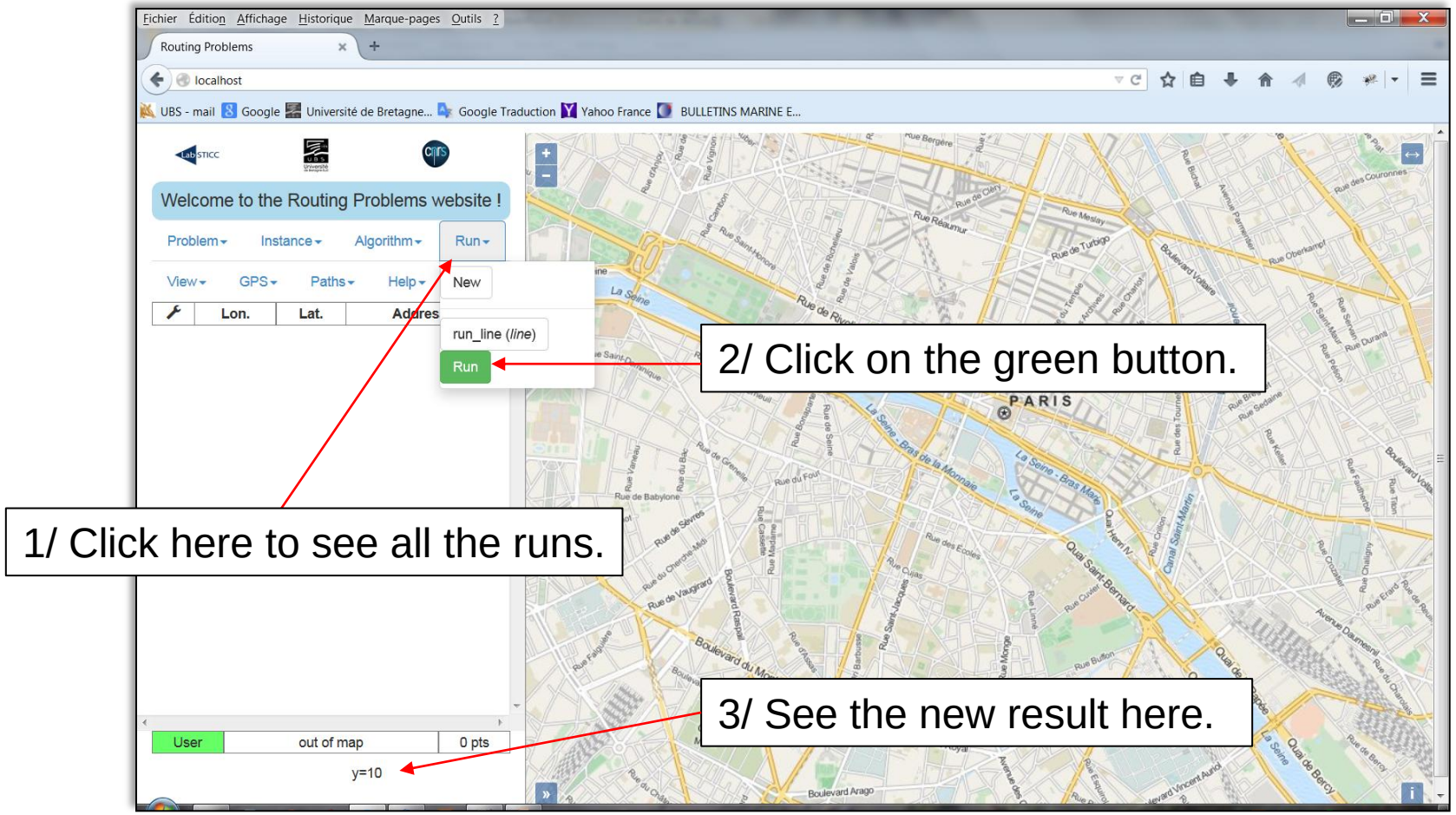
# Execution: launch the web-service and see the result



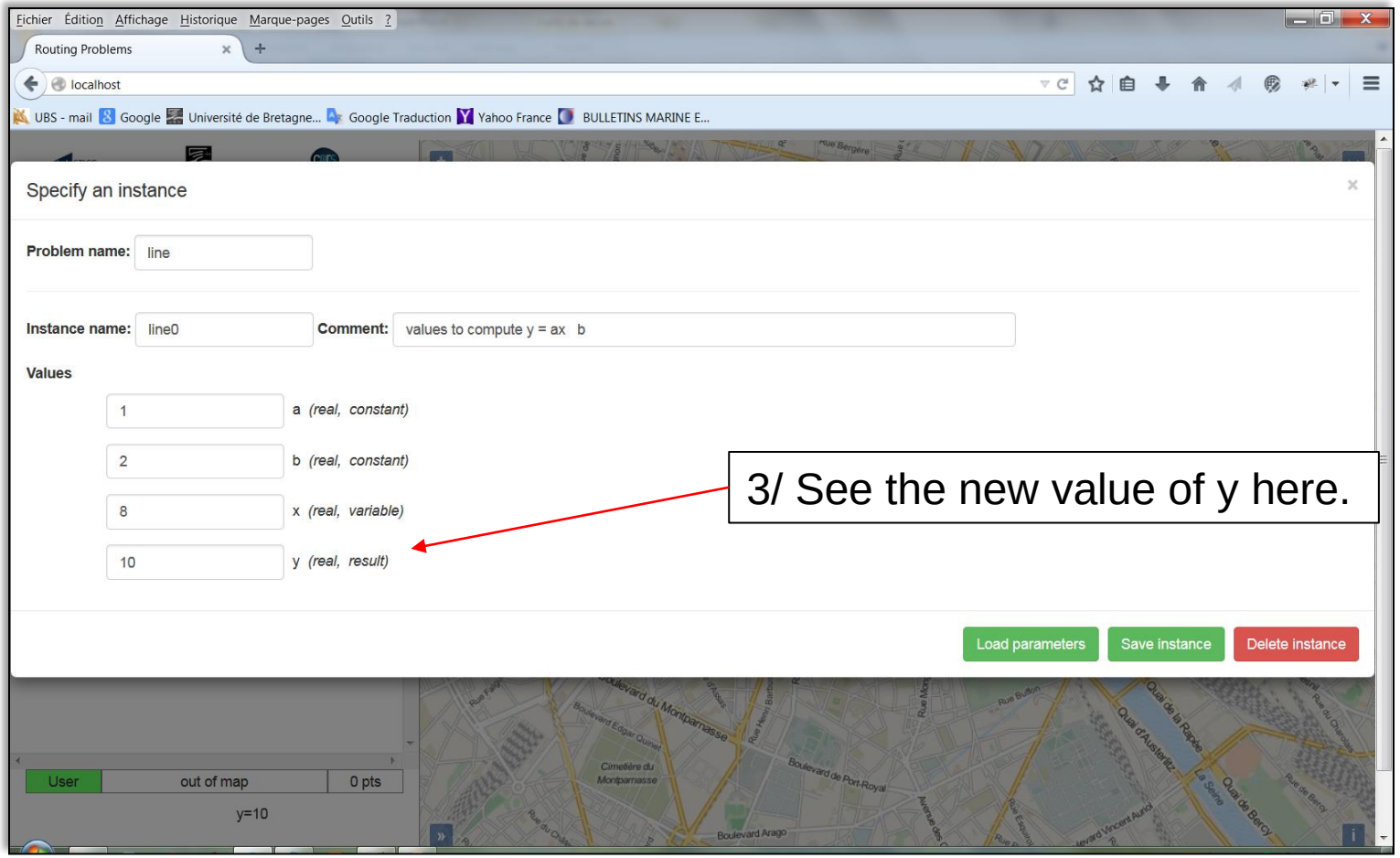
# Execution: change the values of the instance



# Execution: launch the web-service and see the result



# Execution: see the new values in the instance



# Conclusion

Now you know all the basics
to start developing,
installing
and testing
your own web-services.

Congratulations !

You have created your 1rst web-service.