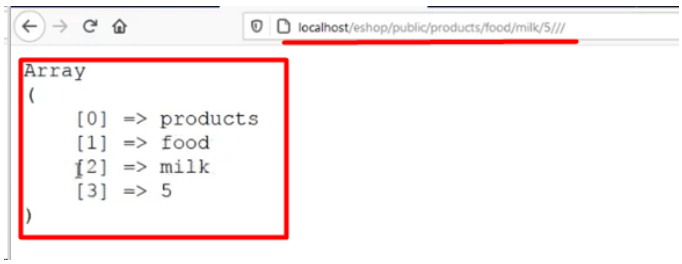
Create your WEB Application with MVC paradigm and OPP – PAP support

Tutorial 3

Considerations: please give maximum attention to the folder and files names because any error name the program don't work.

In continuation from the last tutorial, if you repaired, anything that you've write in url, next to 'localhost/ …. /…../ …..', the result must be equal like the picture.



What is this? So like we see before, the index position number zero is our controller, the index position number one is our method, and the next positions is our parameters.

But we need to divide our url, becouse the program need to get the controller separated from the other things. Equal to method inside the especific controller.

Let's continuos with our code.

Inside the Class app.php now we will write the code like the image, inside the construct that already exists.

```php
$url = $this->parseURL();
show($url);

if(file_exists("../app/controllers/" . strtolower($url[0]) . ".php"))
{

    $this->controller = strtolower($url[0]);
    unset($url[0]);
}

require "../app/controllers/" . $this->controller . ".php";
$this->controller = new $this->controller;

show($url);
```

Next you go to your folder controlers and add a new file a name it home.php.

Inside the new file you will put this code:

```php
<?php

Class Home
{

    public function index()
    {

    }
}
```

If you don't create the file inside you get this faral error:

Warning: require(../app/controlle**j**s/home.phpǰ): failed to open stream: No such file or directory in **C:\xampp\htdocs\eshop\app\core\app.php** on line **22**

Fatal error: require(): Failed opening required '../app/controllers/home.php' (include_path='C:\xampp\php\PEAR') in **C:\xampp\htdocs\eshop\app\core\app.php** on line **22**

Ok, now if you refresh the browser you will see that you have 2 arrays.

The first one is for the first position of the URL, the controller, and the second is for the metohds.

```
Array
(
    [0] => products
    [1] => food
    [2] => milk
    [3] => 5
)
Array
(
    [0] => products
    [1] => food
    [2] => milk
    [3] => 5
)
```

So, in continuation of our routing system code, we now need to create code to check if the method exists and if it does, read the method, otherwise send the user to an error page. To do this, a system is created that checks whether the URL contains a second segment, which is generally used to specify a method within the controller. If the method exists in the controller, it is set as the current method, and the corresponding segment of the URL is removed. This allows you to call specific methods within controllers based on the URL.

```php
//Aqui vamos procurar um metodo dentro do controldaor no array pos [1] da URL
if(isset($url[1]))
{
    //primeiro verificamos se existe um um metodo pos[2] da URL
    $url[1] = strtolower($url[1]);
    if(method_exists($this->controller, $url[1]))
    {
        //remover
        $this->method = $url[1];
        unset($url[1]);
    }
}
```

Now check your browser again with a refresh.

Now we need to get the parameters. Let's now create a variable above the constructor next to the others that already exist.

```php
class App
{
    protected $controller = "home";
    protected $method ="index";
    protected $params;

    //Constructor
    public function __construct()
```

And now, inside the constructor below the existing codo write the code like the picture:

```php
    if (count($url) > 0) {
        $this->params = $url;
    } else {
        $this->params = ["home"];
    }

    call_user_func_array([$this->controller,$this->method],$this->params);
}

//Serve para
private function parseURL()
{
    $url = isset($_GET['url']) ? $_GET['url'] : "home";
    return explode("/", filter_var(trim($url, "/"),FILTER_SANITIZE_URL));
}
```

Let's break the code line by line:

Checking the existence of parameters in the URL: *if (count($url) > 0)*:

This line checks if the $url array contains more than 0 elements. This indicates whether there are parameters in the URL or not. If there is, the condition is true.

Parameter Assignment: If the condition is true (i.e. there are settings in the URL), then $this->params is set to $url, which means the settings from the URL will be used.

If the condition is false (i.e. there are no parameters in the URL), then $this->params is set to ["home"]. This means that a default "home" parameter will be used.

Function Call with Dynamic Parameters:

call_user_func_array ([$this->controller,$this->method],$this->params)

This piece of code calls a function (or object method) dynamically, passing the specified parameters.

$this->controller is the name of the driver to be called.

$this->method is the name of the method within the driver to be called.

$this->params are the parameters to be passed to the method.

Now, if you go to your home.php file inside the controllers folder and put like the picture and reload the browser you will repair that the system routing is already working.

```php
<?php

Class Home
{

    public function index()
    {

        echo "this is the home class inside index method";
    }
}
```

ASSIGMENT: create another controller with a diferent name, and put another message example. That message must have the name of the new controller.

END Tutorial 3

Reference: **PHP Ecommerce website development | Create the app class | MVC OOP - Quick programming**