

# PHP Blog Project

## 8. MVC : les vues (views)

Avant de nous attaquer à la partie View du MVC, nous allons nettoyer le code pour supprimer l'affichage que nous faisons jusqu'à présent.

Nous commentons (ou supprimons) le code ci-dessous dans le fichier index.php

```
8     spl_autoload_register("autoload");
9
10    // $route = filter_var(trim($_SERVER["REQUEST_URI"], '/'), FILTER_SANITIZE_URL);
11    // $routeParts = explode('/', $route);
12    // $controllerName = ucfirst(array_shift($routeParts));
13    // if(empty($controllerName)){
14    //     $controllerName = "Home";
15    // }
16    // echo "Controller Name : $controllerName<br/>";
17    // $actionName = array_shift($routeParts) ?? "index";
18    // echo "Action Name: $actionName<br/>";
19    // $params = $routeParts;
20    // echo "Params : ";
21    // var_dump($params);
22
23    $router = new Controllers\Router();
24    $router->start();
25
```

Ainsi que les echo et var\_dump dans toutes les méthodes "action" de nos contrôleurs, comme dans la méthode index du HomeController ci-dessous

```
4     class HomeController extends BaseController
5     {
6         0 references | 0 overrides
7         public function index(){
8             // echo "<br/>Executing ".get_called_class()." -> ".__FUNCTION__."<br/>";
9             $articleRepository = new ArticleRepository();
10            $articles = $articleRepository->getLastPublishedArticles(12);
11            // var_dump($articles);
12        }
13    }
```

Nous allons maintenant ajouter une méthode render dans notre classe BaseController.

```
app > controllers > BaseController.php > BaseController > render()
3     class BaseController
16
17         1 reference | 0 overrides
18         protected function render($attributes = [], $viewPath = null){
19             extract($attributes);
20             if(!isset($viewPath)){
21                 $controllerName = str_replace("Controller", "", get_called_class());
22                 $controllerName = lcfirst(str_replace("s\\", "", $controllerName));
23                 $viewPath = "views/pages/$controllerName.$this->actionName.php";
24             }
25             require_once $viewPath;
26
27         }
```

2 paramètres :

\$attributes pour importer les entités et éventuellement d'autres paramètres tels que le titre de la page

\$viewPath pour préciser le chemin vers le fichier contenant la vue. Si celui ci n'est pas précisé, la vue devra être stockée suivant un chemin prédéfinie (views/pages/) et porter un nom au format "controllerName.actionName.php"


doc :

<https://www.php.net/manual/en/function.extract.php>

## La view home.index

Nous allons créer notre première vue pour la route / ou /home qui correspond à HomeController -> index()

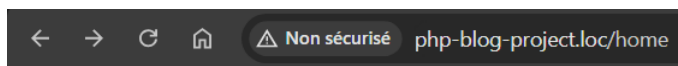
Nous créons donc un fichier home.index.php que nous stockons dans le répertoire views/pages avec dans un premier temps le code ci-dessous

```
app > views > pages >  home.index.php
1  <h1><?= $pageTitle ?></h1>
2
```

Nous allons ensuite modifier la méthode index du HomeController pour appeler la méthode rendre (héritée de BaseController) à laquelle nous passons des attributs comme ci-dessous

```
6  public function index(){
7      // echo "<br/>Executing ".get_called_class()." -> ".__FUNCTION__."()<br/>";
8      $articleRepository = new ArticleRepository();
9      $articles = $articleRepository->getLastPublishedArticles(12);
10     // var_dump($articles);
11     $attributes = [
12         'articles' => $articles,
13         'pageTitle' => "MyBlog - Accueil",
14     ];
15     $this->render($attributes);
16 }
```

Résultat



## MyBlog - Accueil

La variable \$pageTitle de la vue a été remplacée par la valeur que nous avons passée dans le tableau des attributs en paramètre de la méthode render.

Nous pouvons reprendre le code de notre page home.php créé précédemment et le coller dans notre vue en supprimant la partie utilisée pour la connexion à la DB

Pensez à modifier la façon dont vous insérez les articles qui ne sont plus stockés sous forme de tableaux associatifs mais sous forme d'objets (instances de la classe Article)

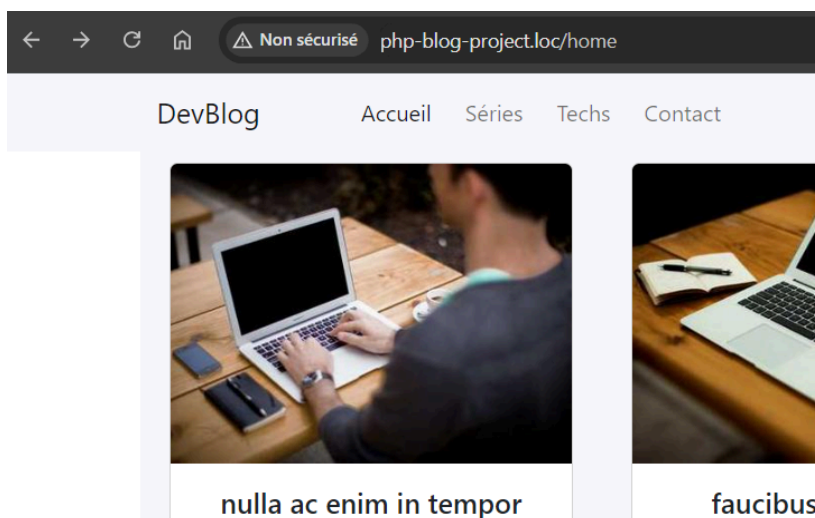
N'oubliez pas de modifier également le lien qui permet d'accéder à la page de consultation d'un article

```

app > views > pages > home.index.php
1  <!DOCTYPE html>
2  <html lang="fr">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title><?= $pageTitle ?></title>
7
8      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet"
9          integrity="sha384-rbsA2VBKQhggwzxH7pPCaQ046MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65" crossorigin="anonymous">
10
11 </head>
12 <body>
13     <div class="container-lg bg-light">
14         <?php include_once "navbar.php"; ?>
15         <main class="mt-5 pt-3 row">
16             <?php foreach($articles as $article){ ?>
17                 <div class="col-12 col-md-6 col-lg-3 d-flex align-items-stretch justify-content-center">
18                     <div class="card mb-3" style="width: 18rem;">
19                         title ?>">
21                         <div class="card-body text-center">
22                             <h5 class="card-title"><?= $article->title ?></h5>
23                             <p class="card-text" style="text-align: justify;">
24                                 <?= $article->summary?>
25                             </p>
26                             <a href="articles/details/<?= $article->id_article ?>" class="btn btn-primary">Lire</a>
27                         </div>
28                     </div>
29                 </div>
30             <?php } ?>
31         </main>
32     </div>
33
34     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"
35         integrity="sha384-kenU1KFdBIE4zVF0s0G1M5b4hpcxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4" crossorigin="anonymous"></script>
36
37 </body>
38
39 </html>

```

## Résultat



Nous en profitons pour déplacer le fichier navbar.php dans une répertoire views/partials et modifions le chemin sur notre view home.index

```
12 <body>
13 <div class="container-lg bg-light">
14     <?php include_once "views/partials/navbar.php"; ?>
15     <main class="mt-5 pt-3 row">
16         <?php foreach($articles as $article){ ?>
```

Nous mettons également à jour les liens de la navbar et déplaçons le fichier contact.php dans views/partials


```
13 <div class="collapse navbar-collapse ms-5" id="navbarContent">
14     <ul class="navbar-nav me-auto mb-2 mb-lg-0">
15         <li class="nav-item me-2">
16             <a class="nav-link active" href="/">Accueil</a>
17         </li>
18         <li class="nav-item me-2">
19             <a class="nav-link" href="/series">Séries</a>
20         </li>
21         <li class="nav-item me-2">
22             <a class="nav-link" href="/techs">Techs</a>
23         </li>
24         <li class="nav-item me-2">
25             <a class="nav-link" href="#" data-bs-toggle="modal"
26         </li>
27     </ul>
28 </div>
29 </div>
30 </nav>
31 <?php include_once 'views/partials/contact.php'; ?>
```

## La view articles.details

Nous modifions la méthode details dans ArticlesController comme nous l'avons fait pour la méthode index dans HomeController en ajoutant les attributs à passer à la vue et en appelant la méthode render à laquelle nous passons les attributs en paramètre.

```
0 references | 0 overrides
7 public function details(){
8     $id = (int)$this->params[0];
9     if($id < 1) {
10         header('HTTP/1.0 404 Not Found');
11         die();
12     }
13     // echo "<br/>Executing ".get_called_class()." -> ".__FUNCTION__;
14     $articleRepository = new ArticleRepository();
15     $article = $articleRepository->getOneById($id);
16     // var_dump($article);
17     $attributes = [
18         'article' => $article,
19         'pageTitle' => "MyBlog - Article : ".$article->title,
20     ];
21     $this->render($attributes);
22 }
```

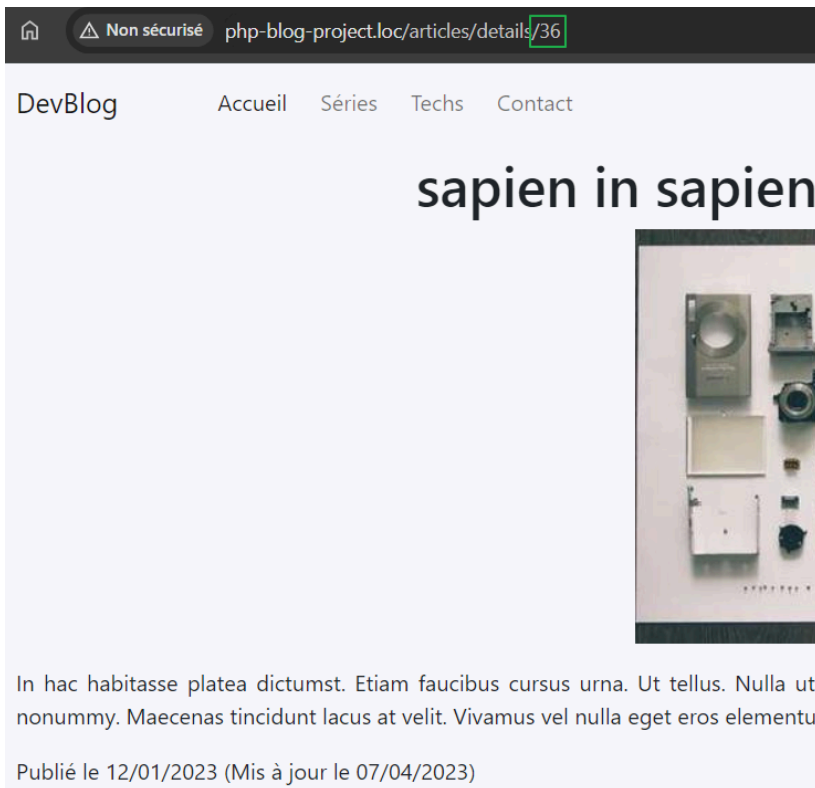
Nous récupérons une partie du code du fichier articles.php créé précédemment pour le coller dans la vue articles.details.php et l'adaptions

```
app > views > pages >  articles.details.php
1  <!DOCTYPE html>
2  <html lang="fr">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title><?= $pageTitle ?></title>
7
8      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" r
9          integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65
10
11 </head>
12 <body>
13     <div class="container-lg bg-light">
14         <?php include_once "views/partials/navbar.php"; ?>
15         <main class="mt-5 pt-3 row">
16             <div class="col-12">
17                 <div class="text-center mb-3">
18                     <h1><?= $article->title ?></h1>
19                     title ?>">
20                 </div>
21                 <p style="text-align: justify;">
22                     <?= $article->summary ?>
23                 </p>
24                 <p>
25                     Publié le <?= date("d/m/Y", strtotime($article->published_at));?>
26                 <?php
27                     if(isset($article->updated_at)){
28                         ?>
29                         (Mis à jour le <?= date("d/m/Y", strtotime($article->updated_at));?>)
30                     <?php
31                     }
32                     ?>
33                 </p>
34             </div>
35         </main>
36     </div>
37
```

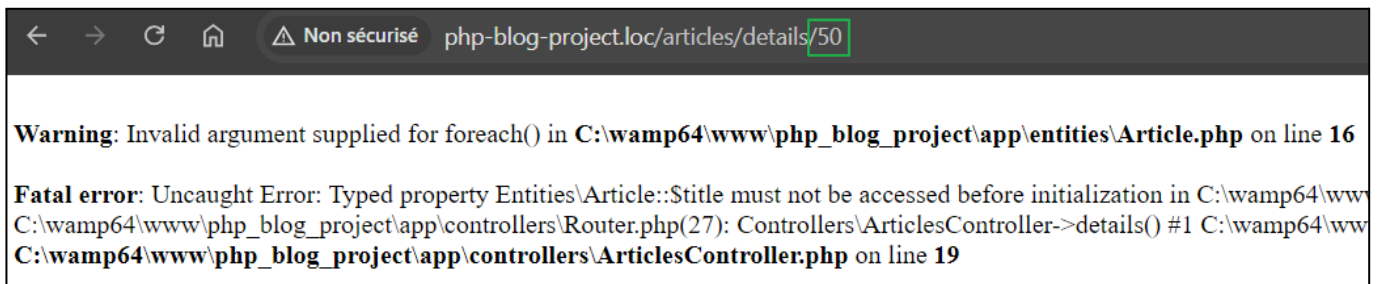
Enfin nous testons tous les cas possibles

-> voir page suivante

## Résultat pour un id existant en DB



## Résultat pour un id n'existant pas en DB



## Résultat pour aucun id passé dans la route



Je vous laisse le soin de corriger ce bug :-)

git :

[https://github.com/DWWM-23526/PHP\\_BLOG\\_PROJECT/tree/Step08](https://github.com/DWWM-23526/PHP_BLOG_PROJECT/tree/Step08)