

`$_SERVER`:

- `$_SERVER` est une table créée par le serveur. elle contient des informations sur le serveur, script de page en cours ainsi l'internaute qui visite la page
- Cette table est une variable prédéfinie donc elle est toujours accessible depuis n'importe quelle classe, fonction ou fichier du site
- Voici une liste non exhaustive des principales clés de la variable superglobale `$_SERVER`

\$_SERVER:

- Voici une liste non exhaustive des principales clés de la variable superglobale \$_SERVER

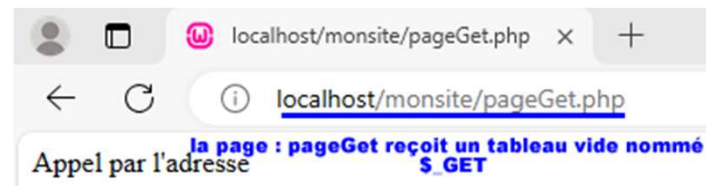
```
$_SERVER["REQUEST_METHOD"]  
Méthode d'appel du script : 'GET', 'HEAD', 'POST', 'PUT'.  
  
$_SERVER["REQUEST_URI"]  
L'url de la page à partir de la racine du site web (contient les paramètres si il existe).  
  
$_SERVER["PHP_SELF"]  
Chemin du script en cours d'exécution à partir de la racine du site web.  
  
$_SERVER["QUERY_STRING"]  
Les paramètres de l'url.
```

- Voir l'exemple:

```
exec.php  
1  <?php  
2      echo $_SERVER['PHP_SELF'].'<br>';  
3      echo '*****<br>';  
4      echo $_SERVER['REQUEST_METHOD'].'<br>';  
5      echo '*****<br>';  
6      echo $_SERVER['REQUEST_URL'].'<br>';  
7  
8  ?>
```

\$_GET:

- \$_GET est une variable superglobale , elle est générée lorsque la page est appelée via la barre d'adresse et reçoit des informations via la barre d'adresse aussi , voir les graphiques ci-dessous:



\$_GET

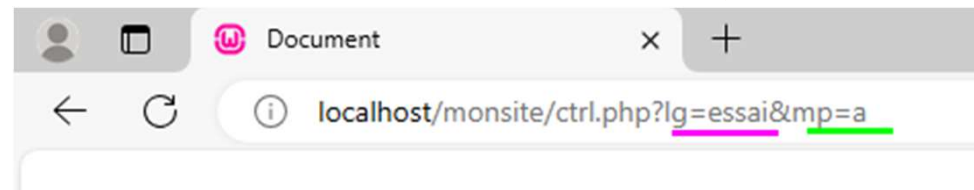
- La table \$_GET est générée aussi via l'envoi des informations par un formulaire ayant comme méthode d'envoi: get. Voir l'exemple

```
<form action="ctrl.php" method="get">
  <p>
    <label for="t1">Introduire votre login</label>
    <input type="text" name="lg" id="t1">
  </p>
  <p>
    <label for="t2">Introduire votre mot de passe</label>
    <input type="password" name="mp" id="t2">
  </p>
  <p>
    <input type="submit" value="OK">
    <input type="reset" value="NO">
  </p>
</form>
```

Exécution

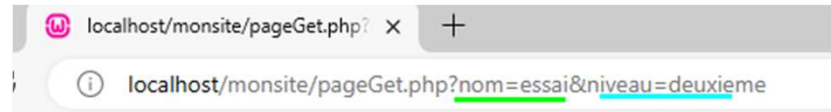
Introduire votre login

Introduire votre mot de passe



RÉCUPÉRATION ET TRAITEMENT DE CONTENU DE LA TABLE \$_GET

- On suppose qu'on a une page PHP nommée pageGet.php, appelée dans le navigateur de cette façon:



- Pour récupérer les paramètres passés en adresse (nom et niveau), la page doit utiliser la table \$_GET qui va contenir deux cases : \$_GET['nom'] et \$_GET['niveau'].
- Voir un exemple de code:

```

1 <?php
2 if($_SERVER['REQUEST_METHOD']=='GET'){
3     if(isset($_GET['nom']) && isset($_GET['niveau']))
4         echo "<br> Bonjour Mr/Mme '$_GET['nom']'. votre niveau est '$_GET['niveau']";
5     else
6         echo "<br> pas de paramètres à afficher!!!!";
7
8
9 }
10
11
12
13 >>
    
```

Tester si la page est appelée via la méthode GET

la méthode isset retourne true, si la case existe

\$_POST:

- Avec la méthode d'envoi GET, les informations transmises à la page sont affichées dans la barre d'adresse. Chose qui est très dangereuse lorsqu'il s'agit des informations sensibles comme login, mot de passe, compte bancaire,..etc.
- C'est pour ça, on doit opter à une autre méthode d'envoi discrète qui est « POST » et qui génère une table superglobale \$_POST dans la page qui va recevoir les informations.
- Voir l'exemple:

MÉTHODE D'ENVOI POST:

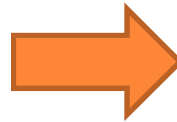
```
<form action="resultat.php" method="post">
  <p>
    Introduire la matricule<input type="text" name="mt">
  </p>
  <p>
    Introduire la nom<input type="text" name="nm">
  </p>
  <p>
    <input type="submit" value="OK">
  </p>
</form>
```



```
resultat.php X
resultat.php
1 <?php
2 echo 'bonjour Mr/Mme '.$_POST['nm'].' votre matricule est '.$_POST['mt'];
3 ?>
```

Introduire la matricule

Introduire la nom



← → ↻ ⓘ localhost/site/resultat.php

bonjour Mr/Mme flen votre matricule est 1

- Après la validation du formulaire, une table nommée \$_POST sera créée automatiquement et envoyée vers la page resultat.php.
- Les cases de la table \$_POST sont indexées par les noms (names) de champs du formulaire

PRÉSENTATION DU PATTERN MVC:

- Le pattern MVC, ou Modèle-Vue-Contrôleur, est un concept fondamental en développement logiciel, largement utilisé pour organiser le code et séparer les préoccupations dans les applications. Voici une petite présentation de ses trois composants:

Modèle (Model) :

- Le modèle représente les données de l'application et la logique métier qui les manipule.
- Il encapsule l'état de l'application et fournit des méthodes pour accéder et manipuler ces données.
- Il ne dépend pas des autres composants du pattern et est donc indépendant de l'interface utilisateur.

PRÉSENTATION DU PATTERN MVC:

Vue (View) :

- La vue est responsable de l'interface utilisateur et de l'affichage des données au sein de l'application.
- Elle représente la présentation des données et interagit avec les utilisateurs.
- Idéalement, elle ne doit pas contenir de logique métier, mais plutôt se concentrer sur l'affichage des informations de manière claire et conviviale.

Contrôleur (Controller) :

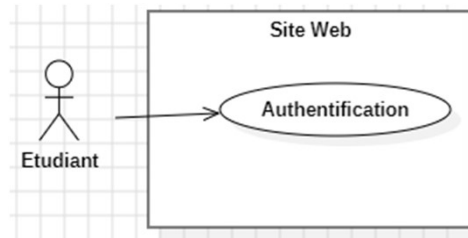
- Le contrôleur agit comme un intermédiaire entre le modèle et la vue.
- Il gère les entrées de l'utilisateur, traite les actions de l'utilisateur et met à jour le modèle en conséquence. Il récupère les données du modèle et les transmet à la vue appropriée pour affichage.

PRÉSENTATION DU PATTERN MVC:

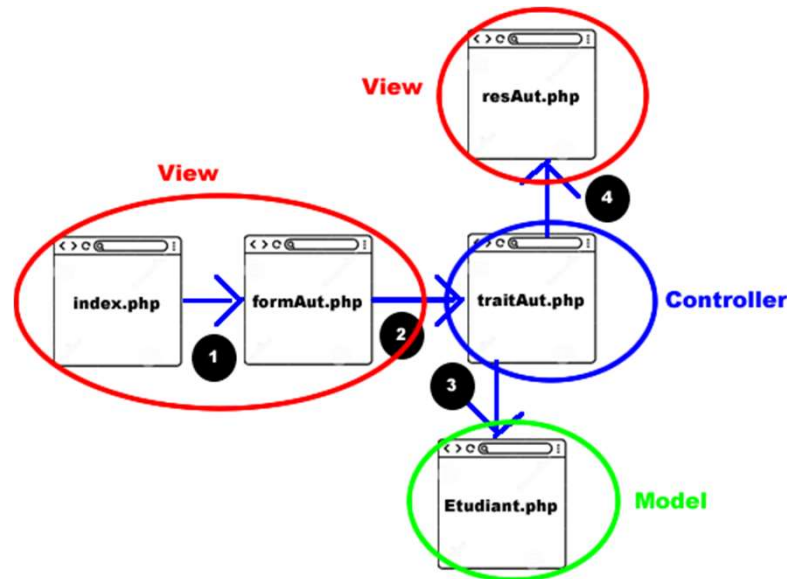
En résumé, le pattern MVC favorise la séparation des préoccupations en organisant le code en trois parties distinctes. Cela rend le code plus modulaire, facile à maintenir et à étendre. De plus, il permet une meilleure collaboration entre les développeurs, car chaque composant peut être développé indépendamment des autres.

APPLICATION DU PATTERN MVC:

- **Exercice:** Implémenter ce cas d'utilisation en PHP tout en respectant le pattern MVC



- Pour résoudre cet exercice, on a créé les page PHP suivantes:

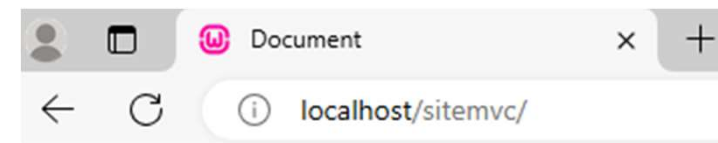


APPLICATION DU PATTERN MVC:

Voici les pages qui répondent à cet exercice:

- Page : index.php → View (contact direct avec l'utilisateur)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <p>
    <a href="formAut.php" target="_blank">Authentifiez vous</a>
  </p>
</body>
</html>
```

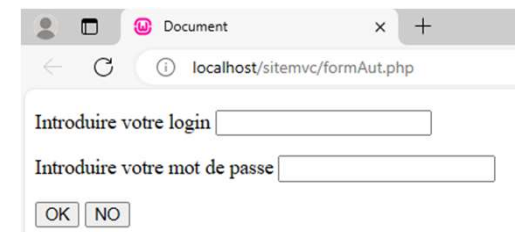


[Authentifier vous](#)

APPLICATION DU PATTERN MVC:

- Page : formAut.php ➔ View (contact direct avec l'utilisateur)

```
formAut.php > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <form action="traitAut.php" method="post">
10     <p>
11       <label for="t1">Introduire votre login</label>
12       <input type="text" name="lg" id="t1">
13     </p>
14     <p>
15       <label for="t2">Introduire votre mot de passe</label>
16       <input type="password" name="mp" id="t2">
17     </p>
18     <p>
19       <input type="submit" value="OK">
20       <input type="reset" value="NO">
21     </p>
22   </form>
23 </body>
24 </html>
```



Document x +

localhost/sitemvc/formAut.php

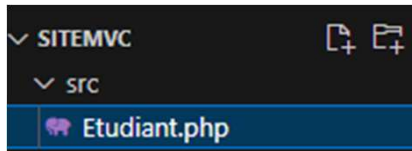
Introduire votre login

Introduire votre mot de passe

OK NO

APPLICATION DU PATTERN MVC:

- Page : Etudiant.php → Model (classes et méthodes) qui va traiter les informations



```
src > Etudiant.php > ...
1  <?php
2      class Etudiant{
3          private $lg;
4          private $mpass;
5          /***** */
6          public function __construct($lg,$mp)
7          {
8              $this->lg=$lg;
9              $this->mpass=$mp;
10         }
11         /***** */
12         public static function verif($lg,$mp){
13             $patLg='^(essai)$/i';
14             $patMp='^(123)$/i';
15             if(preg_match($patLg,$lg) && preg_match($patMp,$mp))return true;
16             else return false;
17         }
18         /***** */
19         public function affiche(){
20             echo 'Félicitation Mr '.$this->lg. ' Vous êtes connectée';
21         }
22     }
23
24  ?>
```

APPLICATION DU PATTERN MVC:

- Page : traitAut.php ➔ Controller (recevoir les informations et désigner le modèle (classe et méthodes) qui va les traiter et les views qui vont dialoguer avec les internautes)

```

1  <?php
2      spl_autoload_register(function($name){
3          require_once('src/'.$name.'.php');
4      });
5
6  >?
7
8  <!DOCTYPE html>
9  <html lang="en">
10 <head>
11     <meta charset="UTF-8">
12     <meta name="viewport" content="width=device-width, initial-scale=1.0">
13     <title>Document</title>
14 </head>
15 <body>
16     <?php
17         if($_SERVER['REQUEST_METHOD']=='GET'){
18             echo 'SVP, passer par le formulaire!!!!';
19             include('formAut.php');
20         }
21         else{
22             $lg=$_POST['lg'];
23             $mp=$_POST['mp'];
24             if(Etudiant::verif($lg,$mp)){
25                 $a=new Etudiant($lg,$mp);
26                 $a->affiche();
27                 include('resAut.php');
28             }
29             else{
30                 echo 'SVP, Vérifiez votre login/mot de passe!!!!';
31                 include('formAut.php');
32             }
33         }
34     }
35 >?
36 </body>
37 </html>

```

Importation des classes

Interdiction de l'accès par adresse

Récupération des données du formulaire

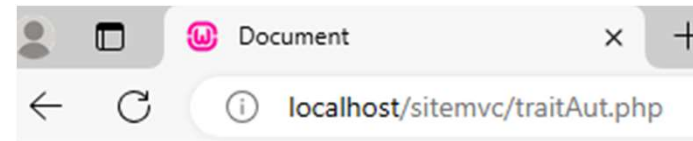
Traitement des données si la saisie est bonne

Traitement si la saisie ne respecte pas les regex

APPLICATION DU PATTERN MVC:

- Page : resAut.php → view (affiche le résultat à l'utilisateur)

```
resAut.php > html > body > p > a
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <p><a href="#" target="_blank"> Autre traitement</a></p>
10 </body>
11 </html>
```

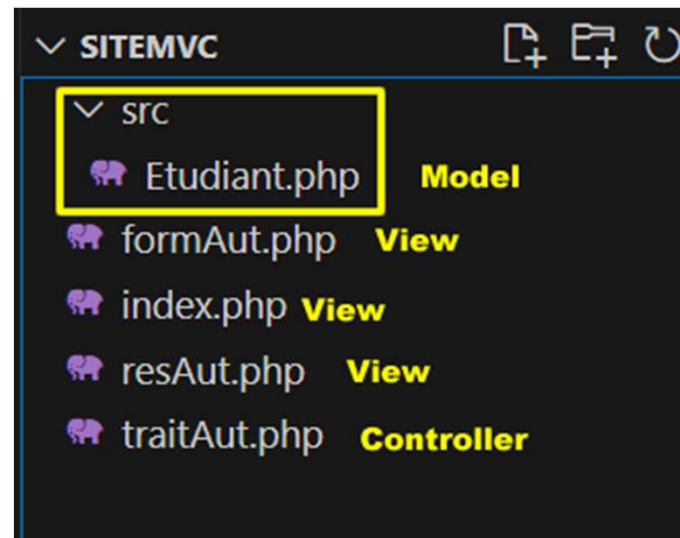


Félicitation Mr essai Vous êtes connectée

[Autre traitement](#)

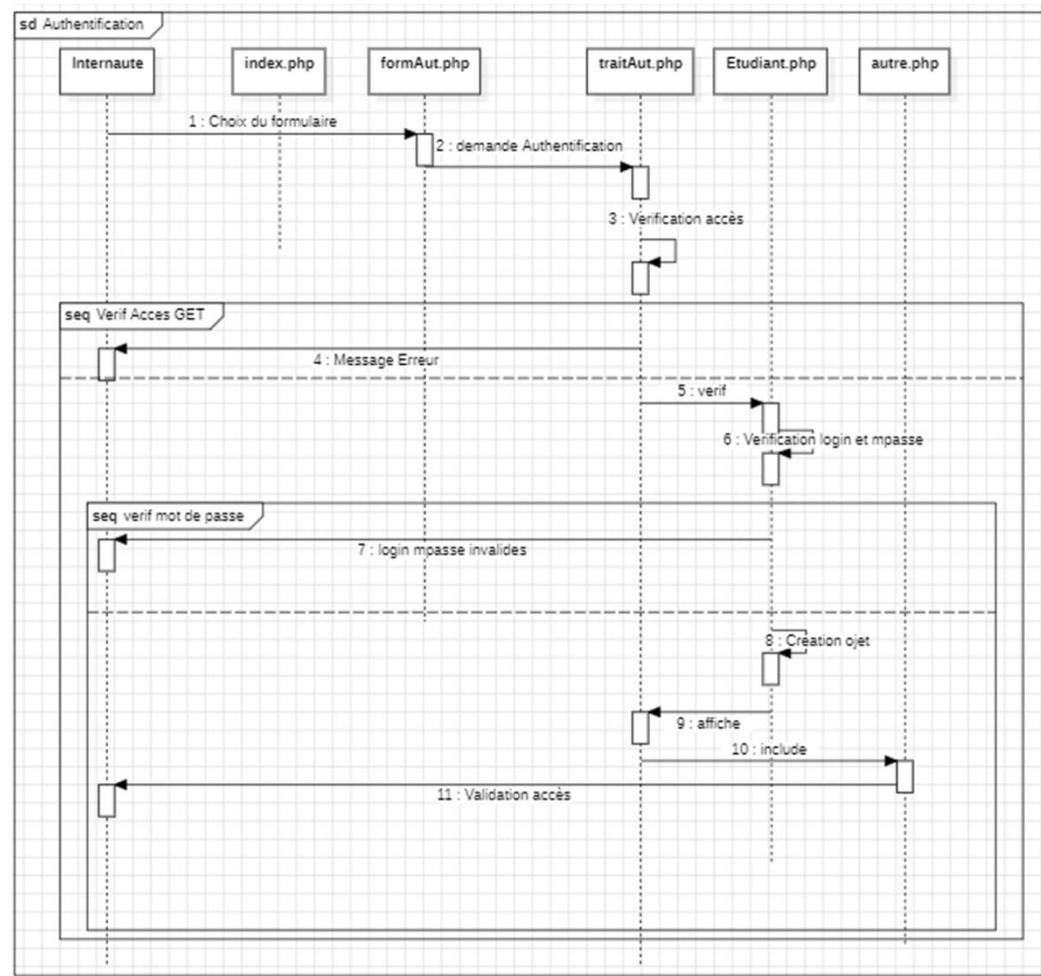
APPLICATION DU PATTERN MVC:

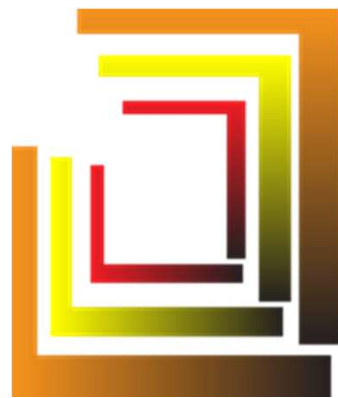
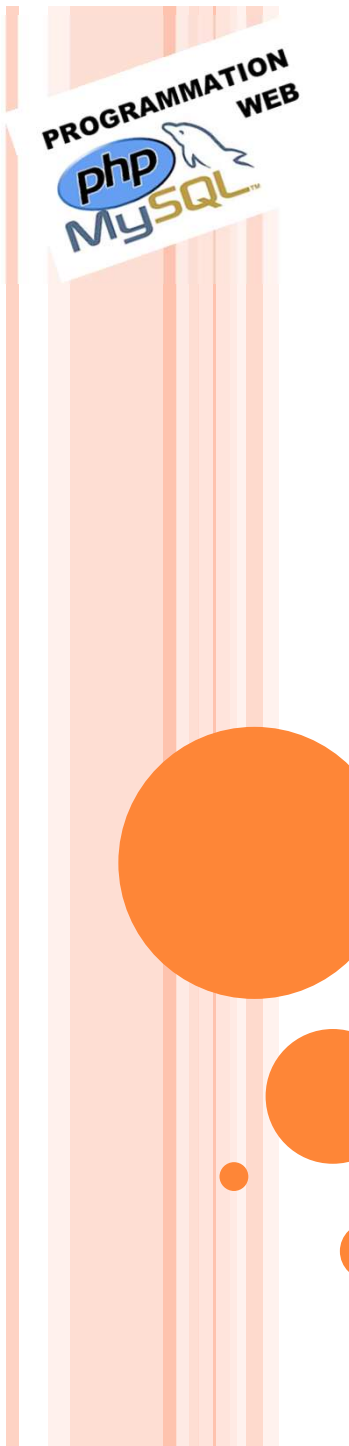
- Voici une vue globale de dossier du site



APPLICATION DU PATTERN MVC:

○ Diagramme de séquence





FIN
Mansour Sihem