

## Fonctionnement du contrôleur principal dans le patron de conception MVC

1.1. Relever les URL en navigant successivement sur les écrans de connexion, de recherche et d'accueil. indiquer quelle partie de l'URL change, en spécifiant le nom de paramètre et la valeur envoyés en méthode GET.

Ecran	URL	Parametre	valeur
connexion	https://localhost/mvc/TP/04-TP%20MVC/RESSOURCE/ressources%20TP5/site/?action=connexion	action	connexion
recherche	https://localhost/mvc/TP/04-TP%20MVC/RESSOURCE/ressources%20TP5/site/?action=recherche	action	recherche
accueil	https://localhost/mvc/TP/04-TP%20MVC/RESSOURCE/ressources%20TP5/site/?action=accueil	action	accueil

1.2. A l'aide de l'annexe 1 et de la réponse à la question précédente, indiquer les valeurs pouvant être prises par la variable \$action.

La fonction `redirigeVers()` visible en annexe 2 est appelée par le contrôleur principal de l'application nommé `index.php`. Cette fonction a un rôle de routage (d'aiguillage). Elle donne accès aux différents contrôleurs de l'application.

Valeurs:

"default"

"liste"

"detail"

"recherche"

"connexion"

"deconnexion"

"profil"

**1.3. Schématiser la structure de la variable \$lesActions créée dans la fonction redirigeVers().**

```
Array(  
    "key" => "value"  
    "key" => "value"  
    "key" => "value"  
    ...  
)
```

**1.4. Quelle partie de cette structure est similaire aux données trouvées en question 1.1 ?**

La clé du tableau

**1.5. À partir de vos réponses aux questions précédentes, indiquer quelles valeurs peuvent être transmises à la fonction redirigeVers() ?**

Une string comportant le nom de la page afin de terminer l'URL

ex:

<https://localhost/mvc/TP/04-TP%20MVC/RESSOURCE/ressources%20TP5/site/> ["liste" => "listeRestos.php"]

**1.6. Pour chacune des valeurs transmises à la fonction indiquer quelle sera la valeur retournée.**

"default" => "listeRestos.php"

"liste" => "listeRestos.php"

"detail" => "detailResto.php"

"recherche" => "rechercheResto.php"

"connexion" => "connexion.php"

"deconnexion" => "deconnexion.php"

"profil" => "monProfil.php"

**1.7. Que se passe-t-il si l'action transmise à la fonction redirigeVers() n'existe pas dans la variable \$lesActions ? Quelle valeur est retournée ?**

`$lesActions["default"] ⇒ "listeRestos.php"`

**1.8. Après avoir consulté l'annexe 3, expliquer à quoi sert la condition utilisant l'instruction `array_key_exists()` dans la fonction `redirigeVers()`.**

La condition sert à vérifier si la clé proposée est valide afin d'afficher la page correspondante.

**1.9. Quel script contrôleur est appelé par `index.php` lorsque la variable GET `action` n'est pas renseignée ?**

`listeRestos.php`

**1.10. Quel script contrôleur est appelé par `index.php` lorsque la variable GET `action` contient le mot clé "liste".**

`listeRestos.php`

## EXERCICE 1 - CGU

Exercice - cf vsCode

**2.1. Repérer dans le menu général l'action correspondant au contrôleur ayant en charge l'affichage des CGU**

**2.2. Placer les fichiers fournis en ressource dans les dossiers appropriés.**

**2.3. Rédiger l'instruction à ajouter à la fonction `redirigeVers()` pour ajouter la nouvelle action dans la variable `$lesActions`.**

**2.4. Ajouter la nouvelle action à la fonction puis tester le bon fonctionnement du lien CGU dans le menu général.**

## Exercice 2 - aimer un restaurant

**3.1. Quelle vue permet d'afficher la fiche descriptive d'un restaurant ?**

vueDetailResto.php

**3.2. Repérer dans le code de la vue le lien correspondant à l'étoile.**

ligne 12

**3.3. Quels sont les paramètres envoyés en méthode GET lorsque l'on clique sur le lien ? Préciser le nom et la valeur de chaque paramètre.**

nom: aimer  
valeur: aimer.php

nom: idR  
valeur: \$unResto['idR']

**3.4. À partir de vos connaissances et du script contrôleur fourni en ressource, déterminer dans quels scripts sont utilisés chacune des deux variables transmises par le lien en méthode GET (celles trouvées à la question précédente).**

aimer.php

**3.5. Placer le fichier fourni en ressource dans le dossier approprié.**

//

**3.6. Rédiger l'instruction à ajouter à la fonction redirigeVers() pour ajouter la nouvelle action dans la variable \$lesActions.**

\$lesActions["aimer"] = "aimer.php";

**3.7. Ajouter la nouvelle action à la fonction puis tester le bon fonctionnement du contrôleur en cliquant sur l'étoile. (Pour tester, il faut être authentifié sur le site)**

Lorsqu'on clique sur l'étoile, on observe que la page affichée reste la même. La page chargée affiche aussi la même URL, pourtant le lien pointé est bien différent.  
//

### 3.8. Le contrôleur `aimer.php` fait-il appel à une vue ?

Non

**3.9. Rechercher sur le web la signification du terme « referer ».**  
**Mettre en commentaire l'instruction `header()` et afficher à la place la valeur de la variable `$_SERVER['HTTP_REFERER']`.**  
**Tester de nouveau l'action d'aimer sur un restaurant.**

En PHP, le terme "referer" fait généralement référence à l'en-tête HTTP `Referer` (oui, il est mal orthographié dans le protocole HTTP, il devrait être "referrer"). Cet en-tête est envoyé par le navigateur web lorsqu'il navigue d'une page à une autre, indiquant l'URL de la page d'origine.

### 3.10. Que contient la variable `$_SERVER['HTTP_REFERER']` ?

Cette variable contient L'URL de la page précédente.

**3.11. Déduire de vos observations le rôle de l'instruction ci-dessous :**  
**`header('Location: ' . $_SERVER['HTTP_REFERER']);`**

Redirection vers la page précédente : L'instruction redirige l'utilisateur vers la page d'où il vient. Cela peut être utile dans divers scénarios, par exemple :

- Après la soumission d'un formulaire, pour revenir à la page précédente.
- Après une action réussie (comme l'ajout d'un commentaire ou la mise à jour d'un profil), pour ramener l'utilisateur à l'endroit où il était.

## **Exercice 3 - inscription**

**4.1 Analyser ce lien, puis à l'aide des fichiers en ressource apporter les modifications nécessaires pour rendre l'inscription fonctionnelle.**