

## Question 1 - Contrôleur de connexion : connexion.php

**1.1. La fonction isLoggedInOn() retourne un booléen, soit true, soit false selon que l'utilisateur est connecté ou pas sur le site.**

**Lors de l'appel à la fonction isLoggedInOn() l'utilisateur est-il connecté par défaut ?**

Non, la fonction isLoggedInOn() vérifie si nous sommes connectés.

**1.2. Quel est le mécanisme sur un navigateur qui rend à TRUE la fonction isLoggedInOn()?**

\$\_SESSION qui stock les info utilisateur.

**1.3. Quelle fonction permet la connexion de l'utilisateur ? Quelle est sa définition (prototype, signature) ?**

Signature = description de la fonction.

Signature : fonction login(string \$mailU, string \$mdpU): void

➔ nom : login

➔ paramètres : \$mailU, \$mdpU

➔ type de retour : void

Prototype = version simple pour tester le fonctionnement.

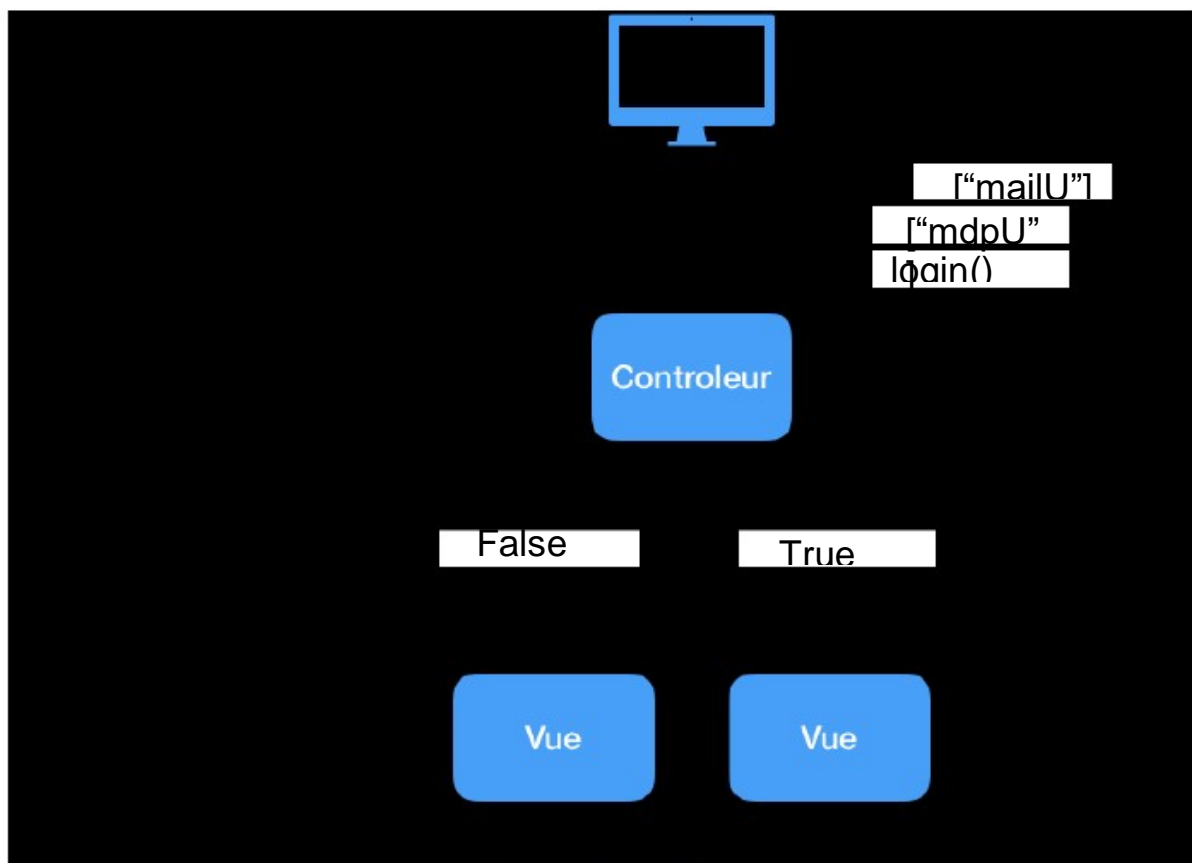
Prototype : if( login(\$mailU, \$mdpU) ) { echo "connecté" ; }

**1.4. À l'aide des questions précédentes et en observant la section de test et le résultat d'exécution du script authentication.inc.php, indiquer le rôle des fonctions suivantes :**

fonction	rôle
login()	Vérifie les informations de connexion
isLoggedInOn()	Vérifie si un utilisateur est connecté
getMailULogged()	Récupère l'adresse mail de l'utilisateur connecté
logout()	Déconnecte l'utilisateur

1.5. À l'aide des annexes 2 et 3 compléter le schéma ci-dessous en indiquant :

- la méthode utilisée pour transmettre les données au contrôleur depuis le formulaire de connexion,
- le nom des indices du tableau associatif qui seront utilisés pour récupérer les données saisies dans le formulaire,
- la valeur retournée par la fonction `isLoggedOn()` orientant l'utilisateur soit sur la vue d'authentification affichant le formulaire de connexion, soit sur la vue de confirmation indiquant que l'utilisateur est bien connecté.



1.6. Compléter le code du contrôleur connexion.php en tenant compte des réponses données précédemment, et en respectant les indications.

CONTROLLER : fichier "connexion.php"

```
//-----  
} else {  
  
    echo "<pre>";  
    print_r($_POST);  
  
    login( $_POST["mailU"] , $_POST["mdpU"] );  
  
    if(isLoggedIn()) {  
        echo "connexion : ok";  
    } else {  
        echo "connexion : non";  
    }  
  
}  
//-----
```

## Question 2 - Contrôleur de déconnexion : deconnexion.php

2.1 Quelle fonction du modèle permet de déconnecter l'utilisateur actuellement connecté sur le site ?

logout()

2.2. Quelle vue permettant de confirmer la déconnexion devra être appelée par ce contrôleur ?

vueDeconnexion.php

2.3. Lors de la déconnexion, est-il utile de transmettre une donnée au contrôleur ?

Oui, pour afficher la vueDeconnexion

**2.4. Compléter le code du contrôleur deconnexion.php. Celui-ci doit appeler la fonction appropriée du modèle, puis afficher à l'utilisateur un message de confirmation conformément aux questions précédentes.**

```
// appel du script de vue qui permet de gerer l'affichage des donnees
$titre = "authentification";
include RACINE . "/vue/entete.html.php";
include RACINE . "/vue/vueAuthentification.php";
include RACINE . "/vue/pied.html.php";
```

### **Question 3 - Analyse et adaptation du contrôleur de recherche : rechercheResto.php**

**3.1. En consultant le script bd.resto.inc.php, indiquer pour chacune de ces trois fonctions leurs signatures (prototype ou définition). Préciser le nom des paramètres attendus en plus de leurs types.**

```
getRestos()
getRestos(): array
```

```
getRestosByNomR($nomR)
getRestosByNomR($nomR:string): array
```

```
getRestosByAdresse($voieAdrR, $cpR, $villeR)
getRestosByAdresse($voieAdrR:string, $cpR:int, $villeR:string): array
```

**3.2. Quel est le contenu de la variable \$\_POST dans les 2 situations suivantes :**

- recherche d'un nom de restaurant
- recherche d'un restaurant selon l'adresse suivante : rue saint remi 33000 bordeaux

1- Array ( [nomR] => charcut ) 1

2-Array ( [villeR] => rue saint remi [cpR] => 33000 [voieAdrR] => bordeaux ) 1

### 3.3. Quels sont les noms de variables transmises au contrôleur en méthode POST lors de la recherche ?

nomR

villeR - cpR - voieAdrR

### 3.4. Compléter la section de récupération des données POST dans le contrôleur afin de faire en sorte que les variables \$nomR, \$voieAdrR, \$cpR et \$villeR soient valorisées correctement en fonction de la recherche effectuée

-----AVANT :

// recuperation des donnees GET, POST, et SESSION

// recherche par nom

\$nomR = null;

// recherche par adresse

\$voieAdrR = null;

\$cpR = null;

\$villeR = null;

-----APRES ; ajouter les post :

\$nomR=null;

if (isset(\$\_POST["nomR"])){

    \$nomR = \$\_POST["nomR"];

}

\$voieAdrR=null;

if (isset(\$\_POST["voieAdrR"])){

    \$voieAdrR = \$\_POST["voieAdrR"];

}

\$cpR=null;

if (isset(\$\_POST["cpR"])){

    \$cpR = \$\_POST["cpR"];

}

\$villeR=null;

if (isset(\$\_POST["villeR"])){

    \$villeR = \$\_POST["villeR"];

}

\$tabIdTC = array();

if(isset(\$\_POST["tabIdTC"])){

    \$tabIdTC = \$\_POST["tabIdTC"];

}

// En ternaire :

\$nomR = isset(\$\_POST["nomR"]) ? \$\_POST["nomR"] : null;

//Etc...

-----AVANT

```
// appel des fonctions permettant de récupérer les données utiles a l'affichage
// Si on provient du formulaire de recherche : $critere indique le type de recherche à effectuer
if (!empty($_POST)) {
    switch ($critere) {
        case 'nom':
            // recherche par nom
            break;
        case 'adresse':
            // recherche par adresse
            break;
    }
}
```

-----APRES ; il faut ajouter \$listeRestos = à la fonction :

```
switch($critere){
    case 'nom':
        // recherche par nom
        $listeRestos = getRestosByNomR($nomR);
        break;
    case 'adresse':
        // recherche par adresse
        $listeRestos = getRestosByAdresse($voieAdrR, $cpR, $villeR);
        break;
}
```

-----AVANT :

```
// appel du script de vue qui permet de gérer l'affichage des données
$titre = "Recherche d'un restaurant";
include RACINE . "/vue/entete.html.php";
include RACINE . "/vue/vueRechercheResto.php";
include RACINE . "/vue/pied.html.php";
```

-----APRES ; il faut ajouter la vue vueResultRecherche.php avec !empty(\$\_POST) :

```
$titre = "Recherche d'un restaurant";
include RACINE . "/vue/entete.html.php";
include RACINE . "/vue/vueRechercheResto.php";
if (!empty($_POST)) {
    // affichage des résultats de la recherche
    include RACINE . "/vue/vueResultRecherche.php";
}
include RACINE . "/vue/pied.html.php";
```

### **3.6. Pourquoi l'appel aux fonctions du modèle est fait lorsque la condition !empty(\$\_POST) est vérifiée ?**

Car elle a besoin de récupérer la valeur contenue.

## **Question 4 - Analyse de la partie existante du contrôleur rechercheResto.php**

**4.1. Dans le code source de la vue, quelle variable permet de choisir l'affichage du formulaire de recherche par nom ou par adresse ?**

`$critere = nom, $critere = adresse`

**4.2. Quelle variable transmise en méthode GET au contrôleur permet de connaître le type de recherche - par nom ou par adresse - que l'on souhaite effectuer ?**

`$_GET["critere"]`, soit `index.php?critere=nom` , `index.php?critere=adresse`

**4.3. Rechercher où est faite cette transmission : quel script ? quelle ligne ?**

rechercheResto.php l'information est récupérée sur le switch (ligne 51 et 55).

ligne 68 lors de l'appel de la vue et à travers le formulaire vueRechercheResto.php

**4.4. Sans cette transmission d'information, le contrôleur pourrait-il savoir quelle recherche effectuer ?**

Non, car `$critere` sera vide.

**4.5. Sans cette transmission d'information, le script de vue pourrait-il savoir quel formulaire afficher ?**

Non plus, car le tableau `$listeRestos` sera vide .

**4.6. Lorsqu'une recherche est effectuée, la vue affiche à nouveau le formulaire de recherche avec des valeurs prédéfinies. Quelles variables sont alors utilisées ?**

`$nomR -> $_POST[' nomR ']`  
`$villeR, $cpR, $voieAdrR -> idem`