BTS SIO SESSION 2019/2020

FICHE technique épreuve E6

Application web d'inscription et de connexion

Dans le cadre de mon stage de première année de BTS SIO, j'ai pu participer au développement d'une application Web, l'application développé par Oxycar. Cette dernière a pour but de proposer un système de mis en place de covoiturage entre les employés d'une même entreprise. Ainsi par le biais de l'application web il est possible de s'inscrire, de se connecter et de visualiser des trajets. Pendant mon stage j'ai travaillé sur la création de ces modules.

Les points clés :

Lorsqu'un utilisateur arrive sur l'application il doit se connecter pour utiliser l'application WEB. Pour se connecter l'utilisateur doit avoir préalablement créé un compte, si ce n'est pas le cas il lui sera possible s'inscrire sur le site Web. Une fois connecté l'utilisateur pourra voir s'afficher des informations propres à lui-même, grâce à un système de session. Aussi il aura la possibilité de se déconnecter pour éviter qu'une autre personne n'utilise son compte. Mon travail a été de créer ces interfaces. Aucune contrainte de d'apparence des interfaces n'a été donnée.

Tables des matières

La table des matières permettra d'ordonnée mon travail.

Une fois terminée et validée par mon maitre de stage, les modules pourront être ajouté à l'application.

1. Présentation du projet

L'objectif de cette partie n'est pas de faire un descriptif complet des technos, ou encore de faire un tutoriel car cela sera trop long et des documentations existent sur internet.

Nous allons voir dans cette partie les différentes technos que j'ai utilisé.

a. Les attentes

Mon projet se divise en plusieurs éléments. Je vais devoir réaliser un formulaire d'inscription fonctionnelle, un formulaire de connexion fonctionnelle, pour cela j'aurais besoin d'une base de données ou les utilisateurs inscrit seront stockés. Les informations demandées seront obligatoires pour pouvoir valider les formulaires, et le stockage du mot de passe devra être crypté. Aussi l'utilisateur devra pouvoir se déconnecter. Pour rendre tous cela accessible il faudra développer des interfaces graphiques.

Une fois connecté les utilisateurs pourront accéder aux données qui les concernes directement sur la page d'accueil.

Par manque de temps cette partie ne sera pas réalisé, mais une simulation de cette dernière sera effectuée.

b. Les technos utilisés

Afin de mener au mieux ce projet j'ai utilisé plusieurs outils qui se complètent les uns des autres.

Pour la partie Frontend j'ai utilisé HTML 5 et CSS 3.

Pour la partie Backend j'ai utilisé PHP 7.4 et SQL 5.7 / SGBB MySQL.

Durant les différentes phases de développement j'ai travaillé sur un serveur local MAMP et utilisé l'IDE Visual Studio Code.

2. Réalisation du projet

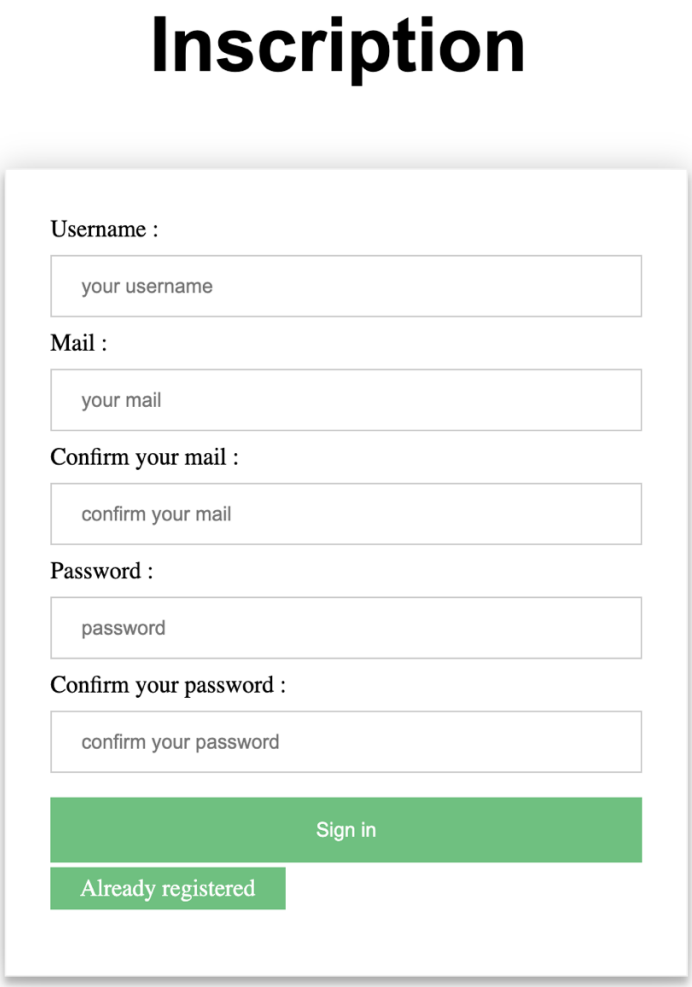
Après avoir fait valider les différentes attentes et mis en place mon espace de travail j'ai pu commencer le développement.

a. L'inscription

Le premier module que j'ai développé fut la partie qui concerne l'inscription d'un utilisateur. Elle est essentielle elle permet de créer un compte sur l'application.

Le Frontend

L'interface créé est représentée ci-dessous :



Cette page présente différents champs.

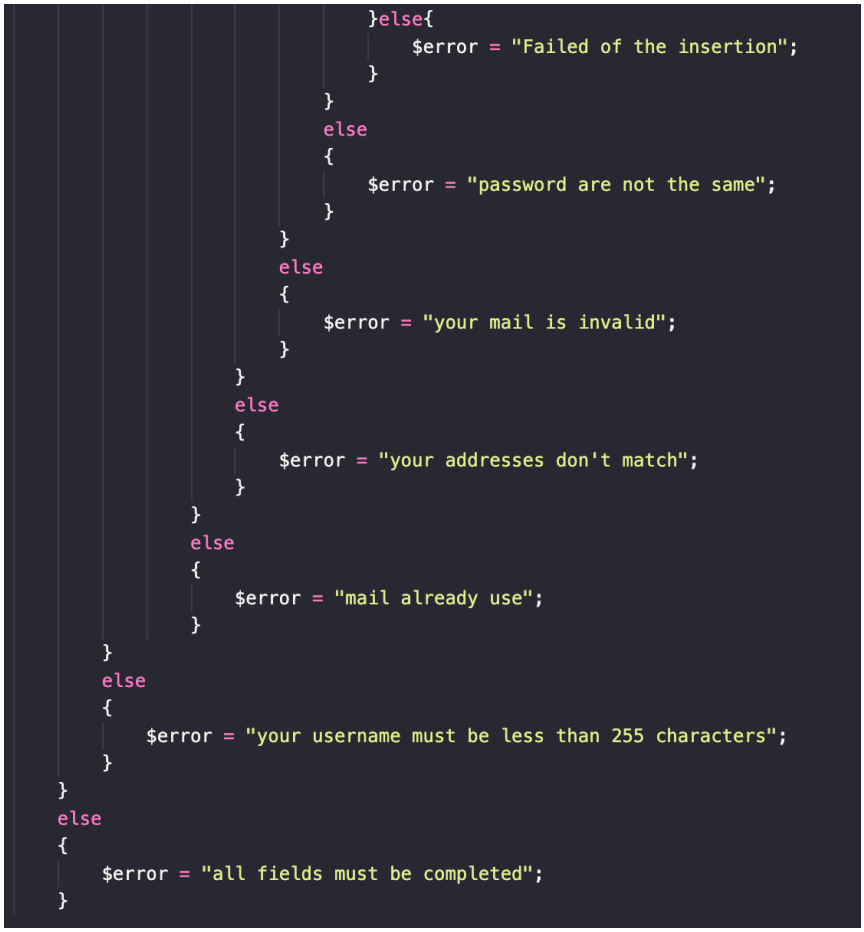
- Un titre : « Inscription »
- Un champ texte pour indiquer le nom de l'utilisateur « your username » ;
- Un champ texte pour indiquer le mail de l'utilisateur « your mail » ;
- Un champ texte pour confirmer le mail de l'utilisateur et éviter une faute dans celui-ci « confirm your mail » ;
- Un champ texte pour indiquer le mot de passe que l'utilisateur devra utiliser pour se connecter « password » ;
- Un champ texte pour confirmer le mot de passe de l'utilisateur et éviter une faute dans celui-ci « confirm your password » ;
- Un bouton de validation du formulaire « Sign in » ;
- Un bouton qui nous envoie sur la page de connexion, au cas au l'utilisateur a déjà un compte « Already registered » ;

L'interface est simple et ne demande pas trop d'information à l'utilisateur car son inscription doit lui permettre de voyager sur l'application, de la quitter et de revenir. S'il souhaite utiliser les fonctionnalités de l'application, qui ne sont pas étudié dans ce document, il devra fournir des informations complémentaires.

Le Backend

Derrière le formulaire présenté ci-dessus, plusieurs vérifications s'effectuent lorsque l'utilisateur le valide. Si un des contrôles est faux alors il renverra un message d'erreur lui correspondant.

Le système d'envoi de message d'erreur :



```
                        }else{
                            $error = "Failed of the insertion";
                        }
                    }
                    else
                    {
                        $error = "password are not the same";
                    }
                }
                else
                {
                    $error = "your mail is invalid";
                }
            }
            else
            {
                $error = "your addresses don't match";
            }
        }
        else
        {
            $error = "mail already use";
        }
    }
    else
    {
        $error = "your username must be less than 255 characters";
    }
}
else
{
    $error = "all fields must be completed";
}
```

Chaque erreur est placée dans un « else » au cas où le « if » correspondant échoue. Ainsi une variable « $error » est instancié avec le message d'erreur correspondant à la raison de l'échec de validation du formulaire.

```php
<?php
if(isset($error))
{
    echo $error;
}
?>
```

Pour afficher ce message à l'utilisateur, une balise PHP apparait si une erreur est détecté et elle affiche le message. Comme présenté ci-contre.

Les différentes vérifications :

Un nombre de six vérifications vont être effectué pour valider le formulaire.

Le premier contrôle si tous les champs ont bien été remplis.
Son script est le suivant :

```php
if(!empty($_POST['username']) && !empty($_POST['mail']) && !empty($_POST['mailComfirm']) && !empty($_POST['pwd']) && !empty($_POST['pwdConfirm']))
```

- Le « ! » avant le terme « empty » signifie la négation, ainsi je vérifie si la valeur n'est pas vide.
- Les termes « && » indique que pour passer à l'étape suivante il faut que les deux côtés du « && » soient correctes.

Si une valeur n'est pas remplie, le formulaire ne pourra pas être validé et le message « all fields must be completed » s'affichera.

Le deuxième contrôle permet de s'assurer que l'utilisateur ne rentre pas un nom d'utilisateur trop grand. La taille pour le nom de l'utilisateur est fixée à 50 caractères.
Son script est le suivant :

```php
$usernameLenght = strlen($username);
if($usernameLenght <= 50 )
```

- La variable « $username » contient le nom qu'a rempli la personne lors de son inscription.
- La fonction « strlen » permet de retourner la taille de la chaine de caractère.
- Le « if » vérifie si la taille de la chaine de caractère est inférieure ou égale à 50.

Si une valeur n'est pas correcte, le formulaire ne pourra pas être validé et le message « your username must be less than 50 characters » s'affichera.

Le troisième contrôle permet de s'assurer que l'utilisateur ne rentre pas un mail déjà utilisé.
Son script est le suivant :

```php
$mail_check = $dbh->connect()->prepare("SELECT * FROM user WHERE mail= ?");
$mail_check->execute(array($mail));
$verif_mail ='';
foreach ($mail_check as $row){$verif_mail=$row['mail'];}
    if ($verif_mail!== $mail)
```

Ce code récupère les mails déjà enregistrés dans la base de données, les compares au mail nouveau mail que rentre le nouvel utilisateur et accepte si le mail n'est pas encore utilisé.
Si une valeur n'est pas correcte, le formulaire ne pourra pas être validé et le message « mail already use» s'affichera.

Le qutrième contrôle permet de s'assurer que l'utilisateur ne se trompe pas dans l'orthographe de son adresse mail.
Son script est le suivant :

```php
if($mail == $mailComfirm)
```

En demandant d'écrire deux fois son mail, l'utilisateur a peu de chance de se tromper dans l'orthographe de son adresse.
Si l'une des valeurs n'est pas correcte, le formulaire ne pourra pas être validé et le message « your addresses don't match» s'affichera.

Le cinquième contrôle permet de s'assurer que l'utilisateur rentre une adresse mail valide, que son format est correct, avec un « @ » et un « . ».
Son script est le suivant :

```php
if(filter_var($mail, FILTER_VALIDATE_EMAIL))
```

La fonction php « filter_var » vérifie ces conditions.
Si une valeur n'est pas correcte, le formulaire ne pourra pas être validé et le message « your mail is invalid» s'affichera.

Le sixième contrôle permet de s'assurer que l'utilisateur ne se trompe pas dans l'orthographe de son mot de passe.
Son script est le suivant :

```
if ($_POST["pwd"] == $_POST["pwdConfirm"])
```

En demandant d'écrire deux fois son mot de passe, l'utilisateur a peu de chance de se tromper dans l'orthographe de ce dernier. Dans cette situation on compare directement les valeurs récupérées sans passer par des variables, car elles stockent le mot de passe mais de manière à ce qu'il soit illisible.

Si une valeur n'est pas correcte, le formulaire ne pourra pas être validé et le message « password are not the same » s'affichera.

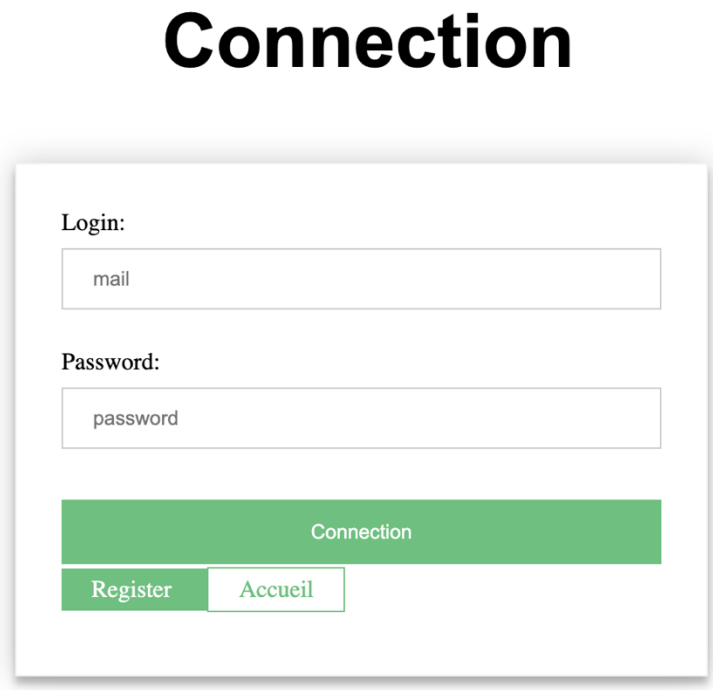
Une fois que tous les champs sont correctement remplis alors les données envoyées par le formulaire pourront être rentré dans la base. L'utilisateur sera renvoyé sur la page de connexion s'il n'y pas de problème, sinon un message d'erreur s'affichera.

b. Connexion / Déconnexion

Après avoir valider le formulaire d'inscription, l'utilisateur pourra se connecter à l'application. Pour cela il lui faudra remplir un autre formulaire, celui de la page de connexion.

Le Frontend

L'interface créé est représentée ci-dessous :



Cette page présente différents champs.

- Un titre : « Connection »
- Un champ texte pour indiquer le mail de l'utilisateur « mail » ;
- Un champ texte pour indiquer le mot de passe que l'utilisateur utilise pour se connecter « password » ;
- Un bouton de validation du formulaire « Connection » ;
- Un bouton qui nous envoie sur la page d'inscription, au cas au l'utilisateur n'a pas de compte « Registere » ;
- Un bouton qui nous envoie sur la page d'accueil, au cas au l'utilisateur ne veut pas se connecter « Accueil » ;

L''interface permet de recueillir les informations nécessaires à la connexion de l'utilisateur, elle permet aussi de se balader entre la page d'inscription et la page d'accueil.

Le Backend

Derrière le formulaire présenté ci-dessus, un système de vérification des valeurs entré accepte la connexion ou pas.
Pour cela il faut récupérer les données de la base afin de les comparer à celles envoyé par le formulaire de connexion.

La connexion à la base se fait par le biais d'un fichier propre à cet effet.

```php
<?php

class Dbh {
    private $servename;
    private $dbUsername;
    private $dbPassword;
    private $dbname;
    private $dbCharset;

    public function connect(){
        $this->servername = "localhost";
        $this->dbUsername = "root";
        $this->dbPassword = "root";
        $this->dbName = "membre";
        $this->dbCharset = "utf8";

        try{
            $dsn = "mysql:host=".$this->servername.";dbname=".$this->dbName.";charset=".$this->dbCharset;
            $pdo = new PDO($dsn, $this->dbUsername, $this->dbPassword);
            $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            return $pdo;
        } catch (PDOException $e) {
            echo "Connection failed: ".$e->getMessage();
        }
    }
}
?>
```

C'est une connexion avec le système de PDO que propose PHP.

Ainsi dans le fichier connxion.php je n'aurais pas à réécrire le script quand je souhaiterai faire une requête sur la base, mais je peux utiliser ce fichier. Pour cela j'utilise le script suivant :

```php
require './loginBDD.php';
    $dbh = new Dbh;
```

Dans un premier temps je vais vérifier que le mail qui sert d'identifiant à l'utilisateur existe déjà dans la base.

```php
$requetuser = $dbh->connect()->prepare("SELECT * FROM user WHERE mail = ? ");

$requetuser->execute(array($mailconnect));
$userexist = $requetuser->rowCount();
if($userexist == 1)
```

Le code ci-dessus contient une requête sur la base, celle-ci récupère les mails existant dans la base de données. Tous les mails sont comparés à celui que l'utilisateur envoie via le formulaire de connexion. Si l'un d'entre eux est identique alors nous passons à l'étape suivante. A l'inverse si aucun mail ne correspond, le message suivant sera affiché : « Wrong mail or password ».

Après avoir le mail, il faut voir si le mot de passe du formulaire lui correspond dans la base de données.

```php
foreach($requetuser as $user) {

    //vérifie son mot de passe
    $passwordChek = password_verify($passwordconnect,$user["pwd"]);
```

Le code ci-dessus vérifie le mot de passe selon le mail avec la fonction PHP « password_verify ». Cette dernière permet de « déhasher » le mot de passe afin de pouvoir le comparer. Évidemment il n'est pas lisible.

Finalement si le mail et le mot de passe correspondant sont correctes alors l'utilisateur peut se connecter. Une fois cela fait, sa session est lancée et l'utilisateur est redirigé vers la page d'accueil.

```php
session_start();
// Ecriture d'une nouvelle valeur dans le tableau de session
$_SESSION['login'] = $user["username"];

// on envoit l'utilisateur vers l'accueil
header("Location:./accueil.php");
```

Si l'utilisateur souhaite se déconnecter un bouton « Logout » est à sa disposition sur la page d'accueil.

Pour déconnecter l'utilisateur il faut détruire sa session, le code est le suivant :

```php
session_start();
$_SESSION = array();
session_destroy();
header('Location:./accueil.php');
```

L'utilisateur est donc renvoyé sur la page d'accueil mais déconnecté.

3. Conclusion

Avec ces deux formulaires l'application va pouvoir gérer chaque utilisateur indépendamment les uns des autres. Ainsi quand l'utilisateur sera connecté il verra les données de sa page d'accueil celle-ci le concernera directement car les données affichées seront propres à l'utilisateur. Il pourra voir les trajets qui lui sont proposés, le conducteur, le prix, la date de départ. Ce système est indispensable pour le bon fonctionnement d'une application.