

Mission réalisée E4 : Création d'un système d'authentification.

<div>Compétences mises en œuvre</div> <div>Réalisations professionnelles (intitulé et liste des documents et productions associés)</div>	Période (sous la forme du JJ/MM/AA au JJ/MM/AA)	Gérer le patrimoine informatique	Répondre aux incidents et aux demandes d'assistance et d'évolution	Développer la présence en ligne de l'organisation	Travailler en mode projet	Mettre à disposition des utilisateurs un service informatique	Organiser son développement personnel et professionnel
		<ul style="list-style-type: none">• Recenser et identifier les ressources numériques• Exploiter des référentiels, normes et standards adoptés par le prestataire informatique• Mettre en place et vérifier les niveaux d'habilitation associés à un service<ul style="list-style-type: none">• Vérifier les conditions de la continuité d'un service informatique• Gérer des sauvegardes• Vérifier le respect des règles d'utilisation des ressources numériques	<ul style="list-style-type: none">• Collecter, suivre et orienter des demandes• Traiter des demandes concernant les services réseau et système, applicatifs• Traiter des demandes concernant les applications	<ul style="list-style-type: none">• Participer à la valorisation de l'image de l'organisation sur les médias numériques en tenant compte du cadre juridique et des enjeux économiques• Référencer les services en ligne de l'organisation et mesurer leur visibilité.• Participer à l'évolution d'un site Web exploitant les données de l'organisation.	<ul style="list-style-type: none">• Analyser les objectifs et les modalités d'organisation d'un projet• Planifier les activités• Évaluer les indicateurs de suivi d'un projet et analyser les écarts	<ul style="list-style-type: none">• Réaliser les tests d'intégration et d'acceptation d'un service<ul style="list-style-type: none">• Déployer un service• Accompagner les utilisateurs dans la mise en place d'un service	<ul style="list-style-type: none">• Mettre en place son environnement d'apprentissage personnel• Mettre en œuvre des outils et stratégies de veille informationnelle<ul style="list-style-type: none">• Gérer son identité professionnelle• Développer son projet professionnel
Réalisation en cours de formation							
Création d'un système d'authentification	18/03/2022 au 23/03/2022					X	

Sommaire :

- **Quelle est ma mission ?**
- **Contexte.**
- **Solutions choisies.**
- **Réalisation de la mission.**
- **Bilan**

Lors de ma formation, la mission que j'ai réalisée est :

- La création d'un système d'authentification.
-

Contexte :

J'ai dû réaliser cette mission pour un site E-commerce. Sur celui-ci il devait comprendre une page connexion et inscription stockant les utilisateurs dans une base de données et avec un mdp haché. Ensuite j'ai dû rendre la connexion obligatoire sur certaines pages du site.

Solutions choisies :

Afin de coder ce service j'ai utilisé les langages HTML / CSS pour le front end, et du PHP pour le backend et SQL pour les requêtes vers la base de données.

Réalisation de la mission :

Premièrement afin de réaliser le système d'authentification j'ai créé un formulaire d'inscription simple :

Connexion Inscription

Prénom

Nom

Email

Téléphone

Mot de passe

Mot de passe

Inscription

Celui-ci a été conçu à l'aide de la balise form :

```
<form action="/app/HTML/inscription_verif.php" method="post">
  <div class="pseudo user_pass">
    <span class="iconify" data-icon="codicon:account"></span>

    <input type="text" placeholder="Prénom" name="prenom" required>

  </div>
  <div class="user_pass">
    <span class="iconify" data-icon="codicon:account"></span>

    <input type="text" placeholder="Nom" name="nom" required>

  </div>

  <div class="user_pass">
    <span class="iconify" data-icon="ci:mail"></span>

    <input class="email" type="email" placeholder="Email" name="email" required>

  </div>
</form>
```

Le principe est lorsque les données du formulaire sont remplies, et que l'utilisateur appuie sur le bouton s'inscrire cela va nous renvoyer vers la page Inscription_verif.php (qui n'est pas voyable).

Sur celle-ci nous allons pouvoir créer des conditions pour les champs.

Exemple :

```
function verifMotDePasse(string $mdp)
{
    if (strlen($mdp) >= 8) {
        $minCarac = false;
        $majCarac = false;
        $specialCarac = false;
        $special = ["!", "*", "#", "$", "&"];

        for ($i = 0; $i < strlen($mdp); $i++) {
            if (ctype_lower($mdp[$i])) {
                $minCarac = true;
            }
            if (ctype_upper($mdp[$i])) {
                $majCarac = true;
            }
            foreach ($special as $caractere) {
                if ($mdp[$i] == $caractere) {
                    $specialCarac = true;
                }
            }
        }
    }

    return $minCarac && $majCarac && $specialCarac;
}
```

Ici, nous avons une fonction afin de donner des conditions au mot de passe.

Le mot de passe devra posséder au minimum 8 caractères. Parmi ces caractères il devra avoir au moins 1 minuscule, 1 majuscule et 1 caractère spécial.

Puis nous réalisons un test :

```
if (verifMotDePasse($password)){  
  
    $cost = ['cost' => 12];  
    $password = password_hash($password, algo: PASSWORD_BCRYPT, $cost);  
  
    $insertion = $db->prepare( query: 'INSERT INTO utilisateur(prenom, nom, email, tel, password) VALUES(:prenom, :nom, :email, :tel, :password)');  
    $insertion->execute(array(  
        'prenom' => $prenom,  
        'nom' => $nom,  
        'email' => $email,  
        'tel' => $tel,  
        'password' => $password,  
    ));  
}
```

Si le mot de passe est bon nous effectuons le reste. (Ici nous cryptons le mot de passe, puis nous insérons les données dans la base de données).

Cet exemple est celui du procédé afin de vérifier le champ 'mot de passe', mais nous faisons au moins un à plusieurs vérifications pour chaque champ.

```
$verifuser = $db->prepare( query: 'SELECT prenom, nom, email, tel, password FROM utilisateur WHERE email = ?');  
$verifuser->execute(array($email));  
$data = $verifuser->fetch();  
$row = $verifuser->rowCount();  
  
$email = strtolower($email);  
if ($row == 0) {  
    if (strlen($prenom) <= 40) {  
        if (strlen($nom) <= 40) {  
            if (strlen($email) <= 75) {  
                if (filter_var($email, filter: FILTER_VALIDATE_EMAIL)) {  
                    if (is_numeric($tel)) {  
                        if ($password === $password_retype) {  
                            // ...  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

Les données du formulaire sont récupérées à l'aide de la méthode \$_POST :

```
if(!empty($_POST['prenom']) && !empty($_POST['nom']) && !empty($_POST['email']) && !empty($_POST['tel']) && !empty($_POST['password']) && !empty($_POST['password_retype'])) {  
  
    $prenom = htmlspecialchars($_POST['prenom']);  
    $nom = htmlspecialchars($_POST['nom']);  
    $email = htmlspecialchars($_POST['email']);  
    $tel = htmlspecialchars($_POST['tel']);  
    $password = htmlspecialchars($_POST['password']);  
    $password_retype = htmlspecialchars($_POST['password_retype']);  
}
```

Nous disons que si les champs sont bien remplis, nous récupérons les données que nous stockons dans des variables.

Puis dans le cas où le formulaire a bien été envoyé ou s'il y'a des erreurs nous faisons une redirection vers la page inscription avec un message :

```

        header( header: 'Location: ../../public/register?reg_err=success');
        die();
    } else {
        header( header: 'Location: ../../public/register?reg_err=password_condition');
        die();
    }
} else {
    header( header: 'Location: ../../public/register?reg_err=password');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=tel_is_int');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=email');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=email_length');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=nom_length');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=prenom_length');
    die();
}
} else {
    header( header: 'Location: ../../public/register?reg_err=already');
    die();
}
}
}

```

```

if (isset($_GET['reg_err'])) {
    $error = htmlspecialchars($_GET['reg_err']);

    switch ($error) {
        case 'success':
            ?>
            <div class="success">
                <strong>Succès</strong> inscription réussie !
            </div>
            <?php
            break;
        case 'password_condition':
            ?>
            <div class="alert ">
                <strong>Erreur</strong> mot de passe ne respecte pas les conditions
            </div>
            <?php
            break;
        case 'password':
            ?>
            <div class="alert ">
                <strong>Erreur</strong> mot de passe différent
            </div>
            <?php
            break;

        case 'email':
            ?>
            <div class="alert ">
                <strong>Erreur</strong> email non valide
            </div>
            <?php
            break;

        case 'email_length':
            ?>
            <div class="alert ">
                <strong>Erreur</strong> email trop long
            </div>

```

En fonction de la situation rencontrée cela nous affichera un message différent.

Au tout début de la page je fais ma connexion à la base de données :

```
try {
    $db = new PDO( dsn: "mysql:host=localhost;dbname=dendo", username: "root", password: "");
    $db->setAttribute( attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);

} catch (PDOException $e) {
    echo "Erreur de la connexion : " . $e->getMessage();
    die();
}
```

Le rendu :

Connexion Inscription

Prénom

Nom

Email

Téléphone

Mot de passe

Mot de passe

Succès inscription réussie !

Inscription

Les données sont stockées dans une table utilisateur :

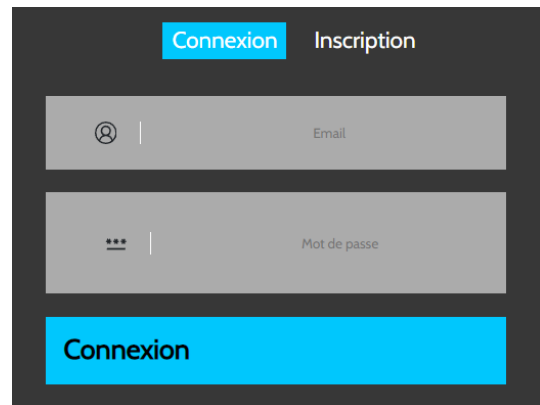
#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1 id	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
<input type="checkbox"/>	2 prenom	varchar(40) utf8_general_ci			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	3 nom	varchar(40) utf8_general_ci			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	4 email	varchar(70) utf8_general_ci			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	5 tel	int(11)			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	6 password	text utf8_general_ci			Non	Aucun(e)			Modifier Supprimer Plus

→

<input type="checkbox"/>	Éditer	Copier	Supprimer	1	Elliot	Guiberteau	test@gmail.com	751976461	\$2y\$12\$4yk3.XcQ7T.i0h2O4ExR0O2yLz3Ha6CEF/XN9aSbHlb...

Ensuite j'ai créé un formulaire de connexion :

Utilisant aussi la balise form renvoyant vers une page de vérification.

The image shows a dark-themed login form. At the top, there are two tabs: 'Connexion' (highlighted in blue) and 'Inscription'. Below the tabs are two input fields: the first is labeled 'Email' with an '@' icon, and the second is labeled 'Mot de passe' with a password icon (three dots). At the bottom of the form is a large blue button labeled 'Connexion'.

Le code de la page :

```
try {
    $db= new PDO( dsn: 'mysql:host=localhost;dbname=dendo;', username: 'root', password: '');
} catch (Exception $e) {
    die("Erreur : " . $e->getMessage());
}

if (isset($_POST['email']) && isset($_POST['password'])) {
    $email = htmlspecialchars($_POST['email']);
    $password = htmlspecialchars($_POST['password']);

    $query = 'SELECT email,password FROM utilisateur WHERE email = ?';
    $verifuser = $db->prepare($query);
    $verifuser->execute(array($email));

    $data = $verifuser->fetch();
    $row = $verifuser->rowCount();

    if ($row == 1) {
        if (filter_var($email, filter: FILTER_VALIDATE_EMAIL)) {
            if (password_verify($password, $data['password'])) {
                session_start();
                $_SESSION['connecte'] =1;
                header( header: 'Location: /public');
                exit();
            } else header( header: 'Location: ../../public/login?login_err=password');
            } else header( header: 'Location: ../../public/login?login_err=email');
        } else header( header: 'Location: ../../public/login?login_err=already');
```

Premièrement je me connecte à ma base de données.

Puis je vérifie que le formulaire a bien été rempli, et ensuite je vérifie si l'email est bien existant dans la base de données, s'il est bien existant je vérifie si l'email et le mot de passe sont bons.

Dans le cas où la connexion s'est bien déroulée, je lance une session, que j'initialise à 1. Afin de dire que l'utilisateur est bien connecté. Puis je redirige l'utilisateur sur la page d'accueil.

Dans le cas où il y'a eu une erreur j'affiche un message d'erreur comme pour la page inscription.

Sur les pages connexion et inscription j'ai rajouté (avec un require du fichier dans le header des pages) :

```
if (connecte()) {  
    header( header: 'Location: /public');  
    exit();}  
?>
```

Qui est une fonction permettant de voir si l'utilisateur est connecté :

```
function connecte (): bool {  
    if (session_status() === PHP_SESSION_NONE) {  
        session_start();  
    }  
    return !empty($_SESSION['connecte']);  
}
```

Si celui-ci est connecté cela nous renverra vers la page d'accueil afin de pas pouvoir revenir sur la page login et register tant qu'on est connecté.

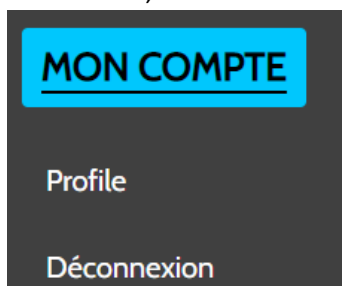
Sur le header afin de se connecter il faut cliquer sur l'icône du bonhomme :



Ou bien dans la partie Mon compte des paramètres :



Lorsque nous sommes connectés cela affiche soit un bouton de déconnexion à la place du bonhomme, soit déconnexion dans l'onglet Mon compte :



La déconnexion est faite à l'aide :

```
<?php
    session_start();
    unset($_SESSION['connecte']);
    header( header: 'Location:login');
    die();
```

Permettant de clear la session connecte.

Finalement, sur une page du site, j'ai fait en sorte que l'on soit obligé d'être connecté afin d'y accéder en mettant ceci :

```
require_once 'connected.php';
connexion_force();
```

Appelant la fonction connexion_force :

```
function connexion_force (): void {
    if(!connecte()) {
        header( header: 'Location: ../public/login');
        exit();
    }
}
```

Faisant en sorte que si nous ne sommes pas connecté cela nous renvoie vers la page login.

Bilan :

Cette mission a été bénéfique pour ma part car elle m'a permis de créer un service informatique, de développer mes compétences en PHP, de pouvoir réaliser des systèmes d'authentications plus efficacement et d'avoir l'expérience afin de rendre mes futurs sites plus sécurisés. Cela m'a aussi permis de travailler en mode projet car cette mission faisait partie d'un plus gros projet étant la création d'un site E-commerce.