Documentation du Projet : TpLoc

1. Introduction

Le projet TpLoc est une application web développée dans le cadre de ma formation en BTS Services Informatiques aux Organisations (SIO). Son objectif principal est de fournir une plateforme permettant aux utilisateurs de se connecter à leur espace personnel de manière sécurisée.

2. Technologies Utilisées

Ce projet repose sur plusieurs technologies clés :

• PHP : Utilisé pour la gestion des sessions et l'authentification des utilisateurs.

• MySQL : Base de données pour stocker les informations des utilisateurs.

• HTML/CSS : Pour concevoir une interface utilisateur intuitive et responsive.

3. Fonctionnalités

L'application intègre les fonctionnalités suivantes :

☑ Connexion des utilisateurs : Authentification via identifiant et mot de passe.

☑ Gestion des erreurs : Affichage de messages en cas d'échec de connexion.

☑ Interface utilisateur conviviale : Une navigation fluide et intuitive.

4. Développement

Étape 1 : Analyse des Besoins

Avant de coder, j'ai défini les besoins des utilisateurs et les fonctionnalités essentielles. Cette analyse m'a permis de structurer le projet et d'anticiper les défis techniques.

Étape 2 : Conception de la Base de Données

J'ai conçu la base de données avec MySQL pour stocker les informations des utilisateurs, notamment :

• Identifiants

• Mots de passe (hashé pour assurer la sécurité)

L'application se connecte en priorité à la base de donnée render, si elle n'est pas disponible elle se connecte alors à la base de donnée en localhost.

```php
// Paramètres pour la base de données Render (PostgreSQL)
$render_dsn = 'pgsql:host=dpg-cv7469jqf0us73au5l90-a.oregon-postgres.render.com;port=5432;dbname=portfolio_database_erzl';
$render_username = 'portfolio_database_erzl_user';
$render_password = '1cHydVkJQ55aH1bOGM2neQOyfQlt0WDZ';

// Paramètres pour la base de données locale (MySQL)
$local_dsn = 'mysql:dbname=tplocation;host=localhost';
$local_username = 'root';
$local_password = '';

try {
    // Essayer de se connecter à Render (PostgreSQL)
    $pdo = new PDO($render_dsn, $render_username, $render_password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connexion réussie à la base de données Render !";
} catch (PDOException $e) {
    // Si la connexion à Render échoue, se connecter à la base de données locale
    try {
        $pdo = new PDO($local_dsn, $local_username, $local_password);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        echo "Connexion à la base de données locale réussie !";
    } catch (PDOException $e) {
        // Si la connexion à la base locale échoue, afficher une erreur
        die("Erreur de connexion : " . $e->getMessage());
    }
}
?>
```

Étape 3 : Développement du protocole de connexion et d'inscription

Le backend est écrit en PHP et gère l'authentification et l'inscription des utilisateurs. Voici un extrait du fichier login.php qui traite la connexion :

```php
if (isset($_POST['connexion'])){
    $identifiant = htmlentities($_POST['identifiant'], ENT_QUOTES, "UTF-8");
    $mdp = htmlentities($_POST['password'], ENT_QUOTES, "UTF-8");
    if (connexion($identifiant,$mdp)){
        header("Location: index.php");
        exit();
    }
    else {
        $error = "Identifiant ou mot de passe incorrect.";
    }
}
```

Vérification de la connexion par rapport à la base de donnée :

```php
function connexion($identifiant, $mdp){
    global $pdo;
    $query = $pdo -> prepare("SELECT user_id, mdp, is_admin FROM utilisateur WHERE identifiant = :identifiant");
    $query->execute(['identifiant' => $identifiant]);
    $user = $query->fetch(PDO::FETCH_ASSOC);

    if ($user && password_verify($mdp, $user['mdp'])) {
        $_SESSION['user_id'] = $user['user_id'];
        $_SESSION['user'] = ($user['is_admin'] == 1) ? 'admin' : 'user';

        return true;
    }
    else {
        return false;
    }
}
```

Vérification de l'inscription en rapport à la base de donnée :

```php
function inscription($identifiant, $mdp, $mail){
    global $pdo;
    $verif = $pdo->prepare("SELECT user_id FROM utilisateur WHERE identifiant = :identifiant OR mail = :mail");
    $verif->execute(['identifiant' => $identifiant, 'mail' => $mail]);

    if ($verif->fetch()){
        return "Cet identifiant ou cet email est déja utilisé.";
    }

    $mdp_hash = password_hash($mdp, PASSWORD_DEFAULT);

    $query = $pdo->prepare("INSERT INTO utilisateur (identifiant, mdp, mail, is_admin) VALUES (:identifiant, :mdp, :mail, 0)");
    $query->execute(['identifiant' => $identifiant, 'mdp' => $mdp_hash, 'mail' => $mail]);
    return true;
}
```

```php
if(isset($_POST['inscription'])){
    $identifiant = htmlentities($_POST['identifiant'], ENT_QUOTES, "UTF-8");
    $mdp = htmlentities($_POST['password'], ENT_QUOTES, "UTF-8");
    $mail = htmlentities($_POST['email'], ENT_QUOTES, "UTF-8");
    if ($_POST['password'] !== $_POST['confirm_password']){
        $error = "Les mots de passe ne correspondent pas.";
    } else {
        $result = inscription($identifiant, $mdp, $mail);
        if ($result === true) {
            $success = "Votre compte a été crée avec succès. Vous pouvez maintenant vous connecter.";
        } else {
            $error = $result;
        }
    }
}
```

Développement du Frontend

L'interface utilisateur a été conçue avec HTML et CSS pour être simple et accessible. Voici un extrait du formulaire de connexion :

```html
<form method="post" action="login.php" class="login-form">
    <div class="form-group">
        <label for="username"><i class="fas fa-user"></i> Identifiant</label>
        <input type="text" id="username" name="identifiant" placeholder="Votre identifiant" required>
    </div>

    <div class="form-group">
        <label for="password"><i class="fas fa-lock"></i> Mot de passe</label>
        <input type="password" id="password" name="password" placeholder="Votre mot de passe" required>
    </div>

    <div class="form-actions">
        <button type="submit" name="connexion" class="btn-login">
            <i class="fas fa-sign-in-alt"></i> Se connecter
        </button>
    </div>
</form>
```

Étape 5 : Tests et Déploiement

Après le développement, j'ai effectué plusieurs tests pour :

• Vérifier le bon fonctionnement de la connexion et l'inscription.

• Tester les erreurs et les cas particuliers.

• Optimiser le code et corriger les bugs.

Une fois l'application stable, je l'ai déployée sur un serveur web.

5. Conclusion

Ce projet m'a permis d'approfondir mes compétences en développement web. J'ai appris à gérer les sessions PHP, sécuriser les données des utilisateurs et concevoir une interface utilisateur ergonomique. Ce fut une expérience enrichissante qui m'a préparé à d'autres projets plus complexes.