DEBUG APPLIFRAIS

Contexte : Pour le BTS SIO SLAM en 1ère année je travaille sur une application de gestion des frais médiaux pour l'entreprise fictive GSB. Cette application est déjà existante mais ne fonctionne pas, le code est vieux, dans un 1er temps je corrige le code en PHP brut et après avoir réparer l'application je la referai entièrement sous Laravel pour plus de performance et modernité.

```php
$idJeuMois = mysqli_query($idConnexion, $req);
if (!$idJeuMois) {
    echo mysqli_error($idConnexion);
}
```

Le principal problème de l'application est qu'elle avais été faite sous une vieille version de PHP, le code utilisé la fonction : mysql_query($req, $idConnexion) ;
Depuis php 7.3 cet fonctionnalité n'existant plus j'ai du remplacer TOUT les mysql_query par des mysqli_query, en inversant les 2 paramètres, ce fût un travail de recherche puis de correction.
Ensuite l'application se lancer mais nous amener sur l'index.

```php
<?php
// Redirige vers la vraie page d'accueil pour éviter de tomber sur l'index
header('Location: cAccueil.php');
exit();
?>
```

J'ai donc créer une page Index.php qui sert de redirection

```php
if ($etape=='validerConnexion') { // un client demande à s'authentifier
    // acquisition des données envoyées, ici login et mot de passe
    $login = lireDonneePost("txtLogin");
    $mdp = lireDonneePost("txtMdp");
    $mdp = md5($mdp); // Permet de hasher les MDP dans la BDD, passé mdp en varchars(32) puis d
    $lgUser = verifierInfosConnexion($idConnexion, $login, $mdp) ;
    // si l'id utilisateur a été trouvé, donc informations fournies sous forme de tableau
    if ( is_array($lgUser) ) {
        affecterInfosConnecte($lgUser["id"], $lgUser["login"]);
    }
    else {
        ajouterErreur($tabErreurs, "Pseudo et/ou mot de passe incorrects");
    }
}
if ( $etape == "validerConnexion" && nbErreurs($tabErreurs) == 0 ) {
    header("Location:cAccueil.php");
}

require($repInclude . "_entete.inc.html");
require($repInclude . "_sommaire.inc.php");

?>
```

J'ai aussi hacher les MDP en md5 (normes de l'époque) pour qu'il ne soit plus stocké en clair dans la BDD, dans la BDD j'ai du augmenter le nombre

de caractères autorisé dans ma colonne mdp à 32.
J'ai créé un formulaire pour envoyer des pièces jointes :

piecejointe.php > html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
<!-- Le type d'encodage des données, enctype, DOIT être spécifié comme ce qui suit -->
<form enctype="multipart/form-data" action="__URL__" method="post">
  <!-- MAX_FILE_SIZE doit précéder le champ input de type file -->
  <input type="hidden" name="MAX_FILE_SIZE" value="30000" />
  <!-- Le nom de l'élément input détermine le nom dans le tableau $_FILES -->
  Envoyez ce fichier : <input name="userfile" type="file" />
  <input type="submit" value="Envoyer le fichier" />
</form>
</body>
</html>
```

J'ai également développé une fonctionnalité qui permet d'envoyer des pièces jointes pour justifier de ces frais hors forfait :

upload.php > a

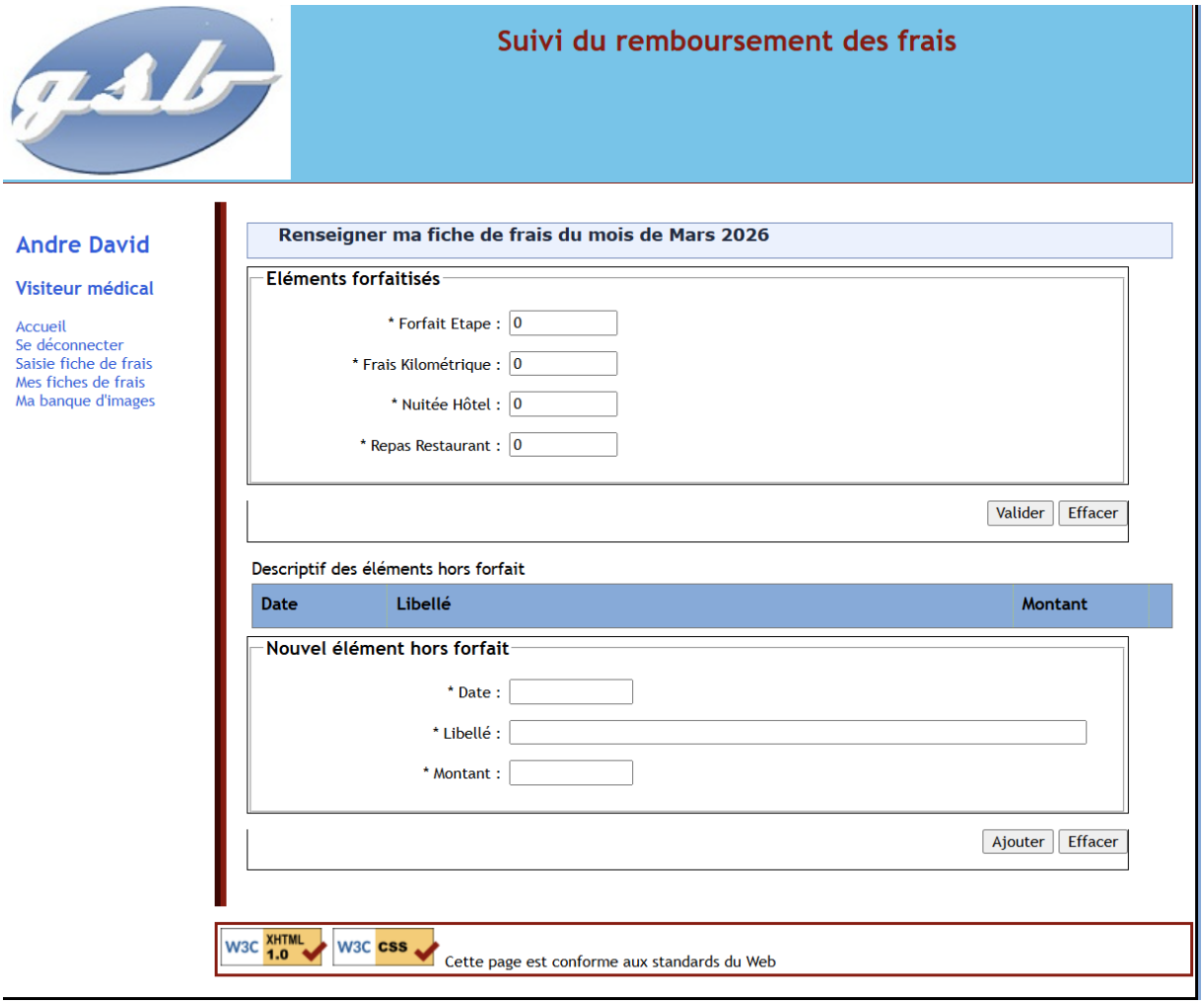
```php
<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_FILES['mon_image'])) {
    $dossier_destination = 'uploads/';

    // Créer le dossier s'il n'existe pas
    if (!is_dir($dossier_destination)) {
        mkdir($dossier_destination, 0777, true);
    }

    $nom_fichier = basename($_FILES['mon_image']['name']);
    $chemin_complet = $dossier_destination . $nom_fichier;

    // Vérification sommaire du type de fichier
    $extension = strtolower(pathinfo($nom_fichier, PATHINFO_EXTENSION));
    $extensions_autorisees = array('jpg', 'jpeg', 'png', 'gif', 'webp');

    if (in_array($extension, $extensions_autorisees)) {
        if (move_uploaded_file($_FILES['mon_image']['tmp_name'], $chemin_complet)) {
            echo "<p style='color: green;'>Succès ! Image ajoutée à la banque.</p>";
        } else {
            echo "<p style='color: red;'>Erreur lors du transfert.</p>";
        }
    } else {
        echo "<p style='color: red;'>Seules les images sont autorisées.</p>";
    }
}
?>
```

Après toute ces modifications l'appli est fonctionnel mais plus aux normes.

Voici l'écran de connexion :

L'écran d'accueil :

Enfin l'écran de suivi des frais :



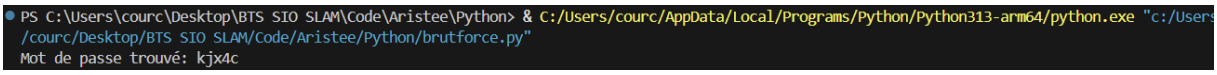
L'application fonctionne mais le design laisse à désirer. Afin de justifier encore plus ma volonté de passé sous laragon j'ai créé un petit script Python qui permet de brutforcer des MDP en md5 :

```python
# Programme de force brute pour trouver un mot de passe de 5 caractères hashé en MD5.
import hashlib

def brute_force_md5(target_hash):
    characters = 'abcdefghijklmnopqrstuvwxyz0123456789'  # Caractères possibles dans le mot de passe
    for a in characters:
        for b in characters:
            for c in characters:
                for d in characters:
                    for e in characters:
                        password = a + b + c + d + e
                        hashed_password = hashlib.md5(password.encode()).hexdigest() # Hashage du mot
                        if hashed_password == target_hash:
                            return password
    return None # Retourne None si aucun mot de passe n'est trouvé
# Exemple d'utilisation
if __name__ == "__main__":
    target_hash = "d07551a62e76546ef39e4c85399f096c" # Hash MD5 du mot de passe "kjx4c"
    found_password = brute_force_md5(target_hash) # Appel de la fonction de force brute pour trouver
    if found_password:
        print(f"Mot de passe trouvé: {found_password}")
    else:
        print("Aucun mot de passe trouvé.")
```

Le MDP n'existe pas dans ma BDD il s'agit d'un mdp de test ressemblant à ceux de la BDD, le hash à été générer avec HashMD5generator.

COUR Cyrille – cyrillecour11@gmail.com – Aristée 2025/2027

Je lance le programme et après 17 seconde :



Le mot de passe a été trouvé.

Maintenant que nous avons une appli fonctionnelle mais graphiquement dépassé et surtout non sécurisé je me tourne vers une option moderne et performante, Laravel le framework PHP le plus populaire.