

DOCUMENTATION TECHNIQUE

Gestion d'intervention



Introduction

La gestion d'interventions est un site WEB permet de gérer les informations relatives aux interventions réalisées pour les clients. L'application est développée en utilisant le langage PHP et la base de données MySQL pour le stockage des données

Configuration

Pour configurer l'environnement de développement et utiliser l'application, j'avais eu besoin des éléments suivants :

1. Serveur distant : pour héberger l'application et la base de données. Il peut s'agir d'un serveur dédié, d'un serveur virtuel ou d'un service d'hébergement.
3. WinSCP : WinSCP est un logiciel de transfert de fichiers sécurisé (SFTP, SCP et FTP) pour Windows. Vous pouvez l'utiliser pour transférer les fichiers de l'application entre votre machine locale et le serveur distant.

Les étapes

Page index envoie vers 3 pages page de recherche d'ajout et de devis



Ajouter

C'est un formulaire pour l'assistance dépannage Ordipointcom.

J'ai commencé par créer le code HTML afin de saisir les informations relatives à chaque intervention, en me basant sur le formulaire papier (informations sur le client, type de service, matériel, etc.). Qui récupère le matériel et le service de la base de données pour une liste décalante

Base de données

La base de données utilisée dans le système de gestion d'interventions s'appelle "testordipointcom". Elle comprend plusieurs tables permettant de stocker les informations nécessaires. Voici un aperçu détaillé de chaque table :

Table "client"

Cette table stocke les informations sur les clients. Voici les colonnes de la table :

- idC : Identifiant unique du client (clé primaire).
- nomC : Nom de famille du client.
- prenomC : Prénom du client.
- telC : Numéro de téléphone du client.
- mail : Adresse e-mail du client.
- adresseC : Adresse du client.
- cpC : Code postal du client.
- villeC : Ville du client.

Table "intervention"

Cette table contient les détails des interventions réalisées. Voici les colonnes de la table :

- idI : Identifiant unique de l'intervention (clé primaire).
- montant : Montant de l'intervention.
- sys_exploitation : Système d'exploitation utilisé lors de l'intervention.
- mdps : Mot de passe associé à l'intervention.
- num_licence : Numéro de licence lié à l'intervention.
- commentaire : Commentaire sur l'intervention.
- idC : Identifiant du client associé à l'intervention (clé étrangère faisant référence à la table "client").

Table "intervention_materiel"

Ce tableau établit une relation plusieurs-à-plusieurs entre les interventions et les matériels utilisés.

Voici les colonnes de la table :

- idIM : Identifiant unique de la relation entre intervention et matériel (clé primaire).
- idI : Identifiant de l'intervention (clé étrangère faisant référence à la table "intervention").
- idM : Identifiant du matériel (clé étrangère faisant référence à la table "materiel").

Table "intervention_service"

Ce tableau établit une relation plusieurs-à-plusieurs entre les interventions et les services associés. Voici les colonnes de la table :

- idIS : Identifiant unique de la relation entre intervention et service (clé primaire).
- idI : Identifiant de l'intervention (clé étrangère faisant référence à la table "intervention").
- idS : Identifiant du service (clé étrangère faisant référence à la table "service").

Table "materiel"

Cette table stocke les informations sur les matériels. Voici les colonnes de la table :

- idM : Identifiant unique du matériel (clé primaire).
- peripheriqueM : Nom du périphérique matériel.

Après avoir créé la base de données sur le serveur local et l'ai liée à mon code PHP. Cependant, j'ai rencontré quelques erreurs que j'ai tenté de corriger. J'ai remarqué des valeurs nulles dans les colonnes "idS" et "idM" de la table "intervention", qui sont des clés étrangères faisant référence aux clés primaires des tables "service" et "matériel". J'ai essayé d'utiliser des jointures dans mes requêtes PHP, mais cela n'a pas fonctionné comme prévu.

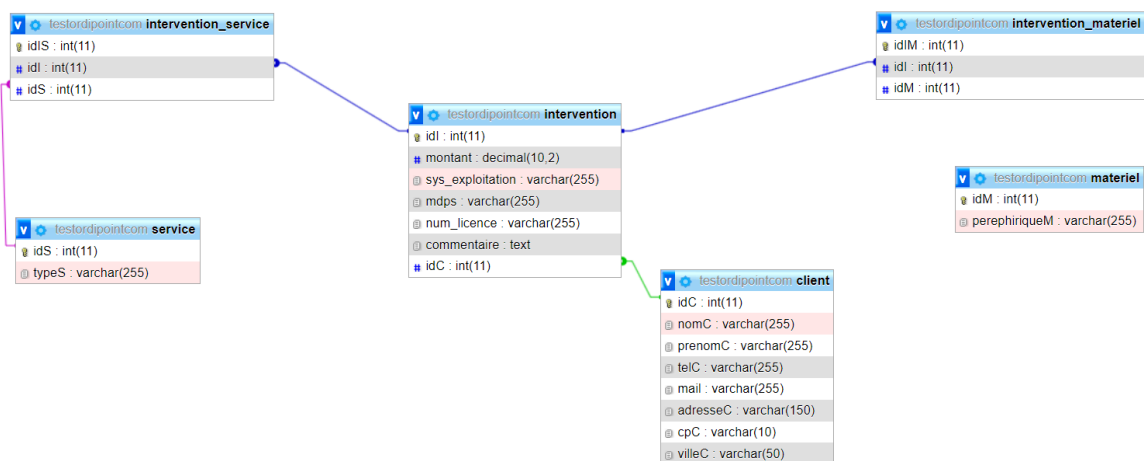
Suite à cela, j'ai décidé de créer une nouvelle table pour relier chaque intervention à son service, ainsi qu'une autre table pour relier chaque intervention à son matériel. J'ai utilisé les requêtes SQL suivantes :

Table "intervention_service" :

```
CREATE TABLE intervention_service (
  idIS INT PRIMARY KEY AUTO_INCREMENT,
  idI INT,
  idS INT,
  FOREIGN KEY (idI) REFERENCES intervention(idI),
  FOREIGN KEY (idS) REFERENCES service(idS)
);
```

Table "intervention_materiel" :

```
CREATE TABLE intervention_materiel (
  idIM INT PRIMARY KEY AUTO_INCREMENT,
  idI INT,
  idM INT,
  FOREIGN KEY (idI) REFERENCES intervention(idI),
  FOREIGN KEY (idM) REFERENCES materiel(idM)
);
```



Malheureusement, cela n'a pas résolu le problème des valeurs nulles. Cependant, les bonnes informations sont bien enregistrées dans la table "intervention_materiel" pour chaque intervention et dans la table "service" pour chaque service.

Pour résoudre ce problème, j'ai décidé de créer une page de confirmation permettant d'afficher toutes les informations enregistrées dans la base de données. J'ai créé une nouvelle page "confirmation.php" et établi un lien entre la page "form.php" (où le code PHP est exécuté lorsqu'on appuie sur "Enregistrer") et la page de traitement. Dans cette page, je commence par vérifier si le client est déjà enregistré dans la base de données :

```
$selectClient = $connex->prepare("SELECT idC FROM client
```

```
WHERE nomC = :nom AND prenomC = :prenom AND telC = :tel AND mail = :mail AND adresseC  
= :adresseC AND cpC = :cpC AND villeC = :villeC");
```

Si le client n'existe pas dans la base de données, j'insère ses informations avec la requête "INSERT INTO" :

```
if (!$existingClient) { $insertClient = $connex->prepare("INSERT INTO client (nomC,  
prenomC, telC, mail, adresseC, cpC, villeC) VALUES (:nom, :prenom, :tel, :mail,  
:adresseC, :cpC, :villeC)");
```

Ensuite, j'utilise la commande suivante pour récupérer l'identifiant du client et pouvoir l'insérer dans les informations d'intervention de la table "intervention" :

```
$clientId = $connex->lastInsertId();
```

J'insère ensuite les informations de l'intervention dans la table "intervention" :

```
$insertIntervention = $connex->prepare("INSERT INTO intervention (montant, sys_exploitation,  
mdps, num_licence, commentaire, idC) VALUES (:montant, :sys_exploitation, :mdps,  
:num_licence, :commentaire, :idC)"); $insertIntervention->bindParam(':montant', $montant);  
$insertIntervention->bindParam(':sys_exploitation', $sys_exploitation);  
$insertIntervention->bindParam(':mdps', $mdps); $insertIntervention->bindParam(':num_licence',  
$num_licence); $insertIntervention->bindParam(':commentaire', $commentaire);  
$insertIntervention->bindParam(':idC', $clientId); $insertIntervention->execute();
```

Au début de mon code, j'ai attribué des valeurs aux différents champs pour faciliter la lisibilité et les modifications :

```
$nom = $_POST['nom'];
```

```
$prenom = $_POST['prenom'];
```

```
$tel = $_POST['tel'];
```

```
$mail = $_POST['mail'];
```

```
$adresseC = $_POST['adresseC'];
```

```
$cpC = $_POST['cpC'];
```

```
$villeC = $_POST['villeC'];
```

```
$montant = $_POST['montant'];
```

```
$sys_exploitation = $_POST['systeme'];
```

```
$mdps = $_POST['mdps'];
```

```
$num_licence = $_POST['licence'];
```

```
$commentaire = $_POST['defauts'];
```

```

$services = $_POST['service'];

$autreService = $_POST['autreS'];

$materiels = $_POST['matériel'];

$autreMatériel = $_POST['autreM'];

```

J'ai fais un href qui permet d'ajouter un nouveau matériel ou service

```

<a href="autreM.php" onclick="ajouterAutreMatériel()">Ajouter autre matériel</a> <!-- Lien
pour ajouter un autre matériel -->

```

Il va vers la page autreM on rajoute le service ensuite on revient et c'est retour et sa garde les informations déjà entrer avec le code suivant

```

<button class="button" onclick="history.go(-2)">Retour</button>

```

J'ai fait la même chose pour service. Après avoir enregistré le code passe par traitement.php pour insérer toutes les valeurs entrer

```

// Vérifier si le formulaire a été soumis
if(isset($_POST['enregistrer'])) {
    // Récupérer les valeurs du formulaire
    $nom = $_POST['nom'];
    $prenom = $_POST['prenom'];
    $tel = $_POST['tel'];
    $mail = $_POST['mail'];
    $adresseC = $_POST['adresseC'];
    $cpC = $_POST['cpC'];
}

```

```

$villeC = $_POST['villeC'];
$montant = $_POST['montant'];
$sys_exploitation = $_POST['systeme'];
$mdps = $_POST['mdps'];
$num_licence = $_POST['licence'];
$commentaire = $_POST['defauts'];
$services = isset($_POST['service']) ? $_POST['service'] : array();
$autreService = $_POST['autreS'];
$materiels = isset($_POST['materiel']) ? $_POST['materiel'] : array();
$autreMateriel = $_POST['autreM'];
    // Vérifier si le client existe déjà dans la base de données
$selectClient = $connex->prepare("SELECT idC
FROM client
WHERE nomC = :nom
AND prenomC = :prenom
AND telC = :tel
AND mail = :mail
AND adresseC=:adresseC
AND cpC=:cpC
AND villeC=:villeC");
$selectClient->bindParam(':nom', $nom);
$selectClient->bindParam(':prenom', $prenom);
$selectClient->bindParam(':tel', $tel);
$selectClient->bindParam(':mail', $mail);
$selectClient->bindParam(':adresseC', $adresseC);
$selectClient->bindParam(':cpC', $cpC);
$selectClient->bindParam(':villeC', $villeC);
$selectClient->execute();
$existingClient = $selectClient->fetch(PDO::FETCH_ASSOC);

// Si le client n'existe pas, l'insérer dans la table "client"
if(!$existingClient) {
    $insertClient = $connex->prepare("INSERT INTO client (nomC, prenomC, telC, mail
, adresseC, cpC, villeC)
VALUES (:nom, :prenom, :tel, :mail, :adresseC, :cpC, :villeC)");
    $insertClient->bindParam(':nom', $nom);
    $insertClient->bindParam(':prenom', $prenom);
    $insertClient->bindParam(':tel', $tel);
    $insertClient->bindParam(':mail', $mail);
    $insertClient->bindParam(':adresseC', $adresseC);
    $insertClient->bindParam(':cpC', $cpC);
    $insertClient->bindParam(':villeC', $villeC);
    $insertClient->execute();
    // Récupérer l'ID du client nouvellement inséré
    $clientId = $connex->lastInsertId();
} else {
    // Utiliser l'ID du client existant
    $clientId = $existingClient['idC'];
}
// Insérer les informations d'intervention dans la table "intervention"


```

```

$insertIntervention = $connex->prepare("INSERT INTO intervention (montant,
sys_exploitation, mdps, num_licence, commentaire, idC)
VALUES (:montant, :sys_exploitation, :mdps, :num_licence, :commentaire, :idC)");
$insertIntervention->bindParam(':montant', $montant);
$insertIntervention->bindParam(':sys_exploitation', $sys_exploitation);
$insertIntervention->bindParam(':mdps', $mdps);
$insertIntervention->bindParam(':num_licence', $num_licence);
$insertIntervention->bindParam(':commentaire', $commentaire);
$insertIntervention->bindParam(':idC', $clientId);
if (!empty($montant)) {
    $insertIntervention->bindParam(':montant', $montant);
} else {
    $insertIntervention->bindValue(':montant', null, PDO::PARAM_NULL);
}
$insertIntervention->execute();

```

La page "confirmation.php" affiche simplement les informations enregistrées. Si l'enregistrement n'a pas été effectué, un titre vide s'affiche à la place.



[Accueil](#)
[Ajouter](#)
[Rechercher](#)

Confirmation de l'intervention

Informations de l'intervention

Numéro de facture: 99

Montant:

Système d'exploitation: win11

Mot de passe: 123

Numéro de licence: aucun

Commentaire:

Informations du client

Nom: fati

Prénom: fafa

Téléphone: 0600000000

Email: fafa@fatima.abd

Adresse: 101

Code postal: 10000

Ville: agen

Services sélectionnés :
Installation

Matériels sélectionnés :
tablette

Modifier

Supprimer

Télécharger

Envoyer

Rechercher

Dans la page de recherche, j'ai ajouté un menu déroulant permettant de choisir le critère de recherche (intervention ou client) ainsi qu'un champ pour saisir les critères de recherche. Les résultats sont affichés dans des boîtes individuelles.

Rechercher par : Valeur de recherche

Si le choix est "intervention", la requête SQL suivante est utilisée pour récupérer les résultats :

```
// Effectuer la recherche en fonction du choix
if ($choix === 'intervention') {
    $sql = "SELECT DISTINCT intervention.idI, intervention.montant,
intervention.sys_exploitation, intervention.mdps, intervention.num_licence,
intervention.commentaire, client.nomC, client.prenomC, client.telC, client.mail
FROM intervention
INNER JOIN client ON client.idC = intervention.idC
INNER JOIN intervention_materiel ON intervention_materiel.idI = intervention.idI
INNER JOIN materiel ON materiel.idM = intervention_materiel.idM
INNER JOIN intervention_service ON intervention_service.idI = intervention.idI
INNER JOIN service ON service.idS = intervention_service.idS
WHERE intervention.sys_exploitation LIKE '%$valeur%'
OR intervention.idI LIKE '%$valeur%'
OR intervention.mdps LIKE '%$valeur%'
OR intervention.num_licence LIKE '%$valeur%'
OR intervention.commentaire LIKE '%$valeur%'
OR client.nomC LIKE '%$valeur%'
OR client.prenomC LIKE '%$valeur%'
OR client.telC LIKE '%$valeur%'
OR client.mail LIKE '%$valeur%'
ORDER BY intervention.idI DESC
";
}
```

Ordre.com Accueil Ajouter Rechercher

Rechercher par : Valeur de recherche

Numéro de facture : 99

Montant :
Système d'exploitation : win11
Mot de passe : 123
Numéro de licence : aucun
Commentaire :
Nom du client : fati
Prénom du client : fafa
Téléphone du client : 060000000
Email du client : fafa@fatima.abd
Matériels :
tablette
Services :
Installation

Numéro de facture : 98

Montant :
Système d'exploitation : hp
Mot de passe : mm
Numéro de licence : ju
Commentaire : m
Nom du client : fati
Prénom du client : fafa
Téléphone du client : 060000000
Email du client : fafa@fatima.abd
Matériels :
Imprimante
Services :
Installation

Numéro de facture : 93

Montant :
Système d'exploitation : Win11
Mot de passe :
Numéro de licence :
Commentaire :
Nom du client : fati
Prénom du client : fafa
Téléphone du client : 060000000
Email du client : fafa@fatima.abd
Matériels :
Tour
Imprimante
Services :
Installation
vente

Numéro de facture : 92 Numéro de facture : 90 Numéro de facture : 89

Dans la recherche client, vous pouvez chercher par nom, prénom, téléphone, e-mail, etc.

Nom du client : fati

Prénom du client : fafa
 Téléphone du client : 060000000
 Email du client : fafa@fatima.abd
 Adresse du client : 202 rue babylone
 Code postal du client : 10000
 Ville du client : La Chaise

[Modifier](#)
[Supprimer](#)
[Ajouter à une intervention](#)

J'ai ajouté un bouton qui permet de récupérer les informations du client directement dans le formulaire sans les retaper

The screenshot shows a web form with three main sections: 'Informations client' (red), 'Informations matériel' (green), and 'Informations service' (orange). The 'Informations client' section contains fields for Nom, Prénom, Téléphone, Email, Adresse, Code postal, and Ville. The 'Informations matériel' section contains a dropdown for Matériel(s), a text field for Système d'exploitation, a text field for Mots de passe, a text field for Numéro de licence, and a text field for Commentaires. The 'Informations service' section contains a dropdown for Services and a text field for Montant. There are also buttons for 'Ajouter autre matériel', 'Ajouter autre Service', and 'Enregistrer'.

Modifier

J'ai ajouté un bouton "Modifier" à la page pour permettre la modification des interventions. J'ai créé la page "modification_intervention.php" pour gérer cette fonctionnalité. J'ai vérifié si l'identifiant de l'intervention était transmis via l'URL et récupéré les informations de l'intervention à partir de la base de données. Ensuite, j'ai affiché un formulaire pré-rempli avec les valeurs actuelles de l'intervention. Lorsque l'utilisateur soumet le formulaire de modification, les modifications sont envoyées à la page "traitement_modifier_intervention.php" pour être traitées et mettre à jour les valeurs de l'intervention dans la base de données.

J'ai ajouté une vérification des données saisies, puis j'ai mis à jour les données de l'intervention à l'aide de la requête UPDATE suivante :

```
$sqlUpdate = "UPDATE intervention SET montant = :montant, sys_exploitation = :sys_exploitation, mdps = :mdps, num_licence = :num_licence, commentaire = :commentaire WHERE idI = :id";
```

Ensuite, j'ai récupéré les données de l'intervention à partir de son ID dans l'URL, ainsi que les données du client, et j'ai affiché l'intervention.

Détails de l'intervention

Numéro de facture: 64
 Montant: 250.00
 Système d'exploitation: Linux
 Mot de passe:
 Numéro de licence:
 Commentaire: Ajouter une carte graphique et une SSD de 500 Go
 Nom du client: ABOUDA
 Prénom du client: fatima
 Téléphone du client: 060004888
 Email du client: fatima@fatima.abd
 Adresse du client: 202 rue babylon, 10000 La Chaise
 Matériel(s): Tour
 Service(s): Installation

J'ai effectué des tâches similaires pour la modification et l'enregistrement des données des clients, ainsi que leur affichage.

Supprimer

J'ai ajouté un nouveau bouton pour la suppression qui récupère l'id de l'intervention et supprime l'intervention avec la requête DELETE. J'ai fait la même chose pour les clients.

```
$interventionQuery = "DELETE FROM intervention WHERE idC = :id";
```

Courrieller

Bouton pour envoyer les informations de l'intervention au client via l'email donner pour faire ceci j'ai dû installer PHPMailer qui est une bibliothèque populaire d'envoi d'email pour PHP pour inclure PHP Mailer j'ai dû installer une composer que j'ai récupérée à partir de ce site

<https://getcomposer.org/>

Après avoir installé le composer j'ai ouvert le terminal on peut aussi le faire avec l'invite de commande j'ai commencé par exécuter cette commande `composer init` ensuite j'ai fait l'installation avec cette commande `composer install` Composer téléchargera PHPMailer et tout autre package et créera un répertoire `vendor` dans mon projet.

J'ai créé une page qui permet l'envoi d'e-mails en utilisant PHPMailer, qui a été installé via Composer. J'ai inclus l'autoloader fourni par Composer en utilisant la ligne suivante :

```
require 'vendor/autoload.php';
```

Cela permet de charger automatiquement PHPMailer et les autres packages requis. Ensuite j'ai fait des recherches pour mieux comprendre comment ça marche

```
$mail->isSMTP(); $mail->Host = 'smtp.example.com';
$mail->Port = 587;
```

Ensuite entrer les information que vous vouliez dans l'email on utilisant ces commande.

Envoyer l'e-mail

```
$mail->setFrom('expediteur@example.com', 'Nom de l'expéditeur');
$mail->addAddress('destinataire@example.com', 'Nom du destinataire'); $mail->Subject
= 'E-mail de test'; $mail->Body = 'Ceci est un e-mail de test';
```

Je n'ai pas pu voir c'est sa as vraiment marcher vu qu'il y avait un problème avec le SMTP de l'entreprise.

Telecharger

j'ai ajouté un nouveau bouton qui permet de télécharger l'intervention au format pdf pour ceci aussi j'avais besoin d'installer une bibliothèque pour faire ceci y'en as plusieurs bibliothèque tcpdf, fpdf etc moi j'ai choisie tcpdf (<https://tcpdf.org/>) je l'ai télécharger à partir de <https://github.com/tecnickcom/tc-lib-pdf> qui montrent aussi les étapes à suivre (le lien pour telecharger FPDF <http://www.fpdf.org/>)

J'ai créé une classe personnalisée appelée "MYPDF" qui étend la classe TCPDF. Cette classe est utilisée pour personnaliser le contenu et la mise en page du PDF généré.

J'ai ajouté du code pour vérifier si l'ID de l'intervention est défini dans la requête GET. Si c'est le cas, j'ai récupéré les informations de l'utilisateur correspondantes à l'ID de l'intervention à partir de la base de données.

Ensuite, j'ai créé une instance de la classe MYPDF que j'avais définie précédemment. J'ai défini les informations du document PDF telles que le créateur, l'auteur, le titre, les mots-clés, les marges et les en-têtes/pieds de page.

J'ai ajouté une nouvelle page au PDF et commencé à construire le contenu du PDF. J'ai affiché les informations générales de l'intervention telles que le nom, le prénom, le téléphone, l'e-mail et l'adresse du client. J'ai également inclus les informations spécifiques à l'intervention telles que le

système d'exploitation, le mot de passe et le numéro de licence, ainsi que les services, le matériel et les commentaires associés à l'intervention, dans un tableau. J'ai également ajouté le montant de l'intervention.

```
class MYPDF extends TCPDF {
    // Page header
    public function Header() {
        // Logo
        $this->Image('ordipointcom/Ordipointcom-Logo.png', 4, 5, 30, '', 'PNG', '',
'T', false, 300, '', false, false, 0, false, false, false);
        // Title and address
        $this->SetFont('helvetica', 'L', 10);
        $this->Cell(270, 5, 'OrdiPointCom', 0, 1, 'C');
        $this->SetFont('helvetica', '', 10);
        $this->Cell(300, 5, '101 Rue de Babylone, 87000 Limoges', 0, 1, 'C');
        $this->Cell(300, 5, '05 55 33 70 20', 0, 1, 'C');
        $this->Cell(300, 5, 'https://ordipointcom.net/', 0, 1, 'C');
        $this->Cell(300, 5, 'info@ordipointcom.net', 0, 1, 'C');

        $this->Ln(10);
    }

    // Page footer
    public function Footer() {
        // Position at 1.5 cm from bottom
        $this->SetY(-15);
        // Font settings
        $this->SetFont('helvetica', 'I', 8);
        // Page number
        $this->Cell(0, 10, 'Page ' . $this->getAliasNumPage() . '/' . $this-
>getAliasNbPages(), 0, 0, 'C');
    }
}
```

Exemple : [intervention.pdf](#)

Serveur sous linux

J'ai ouvert le terminal et exécuté les commandes suivantes pour mettre à jour les paquets de mon système :

```
sudo apt update
```

Ensuite, j'ai installé Apache en exécutant la commande suivante :

```
sudo apt install apache2
```

J'ai également installé PHP en exécutant la commande suivante :

```
sudo apt install php
```

Pour finir, j'ai installé le module PHP pour Apache en utilisant la commande :

```
sudo apt install libapache2-mod-php
```

J'ai transféré tous les fichiers via WinSCP vers le dossier /var/www/html sur le serveur distant. Ensuite, j'ai modifié le fichier bd_connexion.php, qui permet de se connecter au serveur distant.

Voici le contenu modifié du fichier bd_connexion.php :

```

<?php
//Informations de connexion à la base de données
$servername = "nom_du_serveur";
$username = "nom_utilisateur";
$password = "mot_de_passe";
$dbname = "nom_base_de_donnees";
//Connexion à la base de données
$conn = new mysqli($servername, $username, $password, $dbname);
//Vérification de la connexion
if ($conn->connect_error) {
die("La connexion à la base de données a échoué : " . $conn->connect_error);
} else {
echo "Connexion à la base de données réussie !";
}
//Fermeture de la connexion
$conn->close();
?>

```

J'ai rencontré un problème lors de la saisie des informations. Lorsque je ne saisis rien dans le champ "montant", l'intervention ne s'enregistre pas et une erreur 500 est affichée. Pour résoudre ce problème, j'ai ajouté la condition suivante pour le champ "montant":

```
if (empty($montant)) { $montant = null; }
```

Cela permet de définir la valeur de la variable \$montant sur null lorsque le champ est vide, ce qui permet d'enregistrer l'intervention correctement. J'espère que cette rédaction répond à vos attentes. N'hésitez pas à me demander si vous avez d'autres questions ou besoin d'aide supplémentaire !

Devis

Lorsque le formulaire est soumis en cliquant sur le bouton "Générer le devis", les informations saisies sont récupérées à l'aide de la superglobale `$_POST`. Ensuite, le contenu du devis est généré au format HTML en utilisant les informations récupérées.

Le contenu du devis est ensuite utilisé pour créer un fichier PDF à l'aide de la bibliothèque TCPDF. Le fichier PDF est généré avec les informations du devis et affiché dans le navigateur pour être téléchargé par l'utilisateur.

Le formulaire comporte également une section pour les informations du client, les informations matérielles (qui peuvent être ajoutées dynamiquement) et les informations de facturation.

La section de style CSS dans le code est utilisée pour appliquer un style visuel au formulaire.

Veuillez noter que ce code nécessite la présence de la bibliothèque TCPDF (fichiers inclus dans le répertoire `TCPDF-main`) pour fonctionner correctement. Assurez-vous de disposer de ces fichiers pour exécuter le code sans erreur.