Structure et configuration des fichiers

Structure du dossier

```
\-- phpcrud
   |-- index.php
   |-- create.php
   |-- read.php
   |-- update.php
   |-- delete.php
   |-- functions.php
   |-- style.css
```

Ce que chaque fichier contiendra :

index.php : Page d'accueil de notre application CRUD.
create.php : Créez de nouveaux enregistrements avec un formulaire HTML et envoyez des données au serveur avec une requête POST.
read.php : Affichez les enregistrements de notre table de base de données et naviguez avec la pagination.
update.php : Met à jour les enregistrements existants avec un formulaire HTML et envoie les données au serveur avec une requête POST.
delete.php : Confirme et supprime les enregistrements par ID (requête GET pour obtenir l'ID).
functions.php : Fonctions de base de modèles et fonction de connexion MySQL (nous n'avons donc pas à répéter le code dans chaque fichier).
style.css : La feuille de style de notre application, cela changera l'apparence de notre application.

Création de la base de données et configuration des tables
Créer une base de données, entrez phpcrud et sélectionnez utf8_general_ci comme collation

Le code situe dans database.txt SQL créera la table : contacts , nous utiliserons cette table dans notre application.

Création de la feuille de style (CSS3)

Le fichier se nomme style.css
N'hésitez pas à changer le style, c'est ce que j'ai mis en place pour rendre l'application CRUD plus attrayante.

Création de l'application CRUD

Création des fonctions

Ce fichier contiendra des fonctions que nous pourrons exécuter dans tous nos fichiers PHP, c'est pour ne pas avoir à écrire le même code dans chaque fichier PHP, plus le code est court, mieux c'est, n'est-ce pas ? Nous allons créer 3 fonctions, 1 fonction se connectera à la base de données, les 2 autres seront les modèles pour l'en-tête et le pied de page qui apparaîtront sur chaque page que nous créerons et contiendront la mise en page HTML.

Le fichier se nomme functions.php

Création de la page d'accueil

index.php sera notre page d'accueil.

Cela créera une page d'accueil de base, nous pouvons utiliser cette page pour naviguer vers les autres pages. Comme vous pouvez le voir, nous incluons le fichier functions.php et exécutons les fonctions de modèle que nous avons créées, rappelez-vous que ces fonctions ajouteront le code d'en-tête et de pied de page à notre page d'accueil.

Création de la page de lecture

Cette page remplira les enregistrements de notre table de contacts dans une table HTML.
Le fichier se nomme read.php

```
<?php
include 'functions.php';
// Connect to MySQL database
$pdo = pdo_connect_mysql();
// Get the page via GET request (URL param: page), if non exists default the page to 1
$page = isset($_GET['page']) && is_numeric($_GET['page']) ? (int)$_GET['page'] : 1;
// Number of records to show on each page
$records_per_page = 5;
```

Une fois de plus, nous incluons le fichier de fonctions, mais cette fois nous nous connectons à notre base de données MySQL en exécutant la fonction : pdo_connect_mysql, si la connexion réussit, nous pouvons utiliser le $pdo variable pour exécuter des requêtes.

Nous créons également 2 autres variables, la $page déterminera la page sur laquelle l'utilisateur se trouve actuellement, la $records_per_page sera utilisé pour limiter le nombre d'enregistrements à afficher sur chaque page, par exemple, si nous limitons le nombre d'enregistrements à 5 et que nous avons 10 enregistrements dans notre table de contacts , alors il n'y aura que 2 pages et 5 enregistrements sur chaque page , l'utilisateur pourra naviguer entre les pages.

Le code suivant

```
// Prepare the SQL statement and get records from our contacts table, LIMIT will determine the page
$stmt = $pdo->prepare('SELECT * FROM contacts ORDER BY id LIMIT :current_page, :record_per_page');
$stmt->bindValue(':current_page', ($page-1)*$records_per_page, PDO::PARAM_INT);
$stmt->bindValue(':record_per_page', $records_per_page, PDO::PARAM_INT);
$stmt->execute();
// Fetch the records so we can display them in our template.
$contacts = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

Le code ci-dessus sélectionnera les enregistrements de la table des contacts , cela sera déterminé par la page actuelle sur laquelle se trouve l'utilisateur, les enregistrements seront triés par la colonne id.

Nous utilisons également une instruction préparée pour la requête ci-dessus, cela garantira que notre requête est sécurisée (échappe aux données d'entrée de l'utilisateur).

```
// Get the total number of contacts, this is so we can determine whether there should be a next and previous button
```

```
$num_contacts = $pdo->query('SELECT COUNT(*) FROM contacts')->fetchColumn();
?>
```

La requête SQL ci-dessus obtiendra le nombre total d'enregistrements dans la table des contacts , nous n'avons pas besoin d'utiliser une instruction préparée ici car la requête n'inclut pas de variables d'entrée utilisateur.

Le code situé entre ces deux balises de la page read.php

```
<?=template_header('Read')?>

<?=template_footer()?>
```

est le modèle de la page de lecture, le code itère les contacts et les ajoute au tableau HTML, nous pourrons lire les enregistrements sous forme de tableau lorsque nous naviguerons vers la page de lecture.

La pagination est ajoutée afin que nous puissions naviguer entre les pages de la page lue (page 1, page 2, etc.).

Pour les icônes que nous utilisons Font Awesome , assurez-vous qu'il est inclus dans la fonction de modèle d'en-tête ou les icônes n'apparaîtront pas.

La page de création

La page de création sera utilisée pour créer de nouveaux enregistrements et les insérer dans notre table Contacts .

Le code de la page vérifiera si le tableau POST (données du formulaire) n'est pas vide, si ce n'est pas le cas, cela signifie essentiellement que l'utilisateur a rempli le formulaire et cliqué sur le bouton Soumettre, cela insérera alors un nouvel enregistrement dans notre table Contacts .

```
<?=template_header('Create')?>

<?=template_footer()?>
```

Entre ces balises le modèle de notre page de création, création d’un formulaire et nommé chaque champ de saisie en conséquence, le nom du champ de saisie est la façon dont nous obtiendrons la variable POST dans notre code PHP, par exemple, si nous nommez un champ de saisie "zip_code", nous pouvons obtenir la valeur de ce champ de saisie avec $_POST['code_postal']en PHP (en supposant que la méthode du formulaire est définie sur post).

La page de mise à jour

La page de mise à jour sera utilisée pour mettre à jour les enregistrements dans notre table Contacts , cette page est similaire à la page de création mais au lieu d'insérer un nouvel enregistrement, nous mettrons à jour les enregistrements existants. Nous pourrons obtenir l'ID d'enregistrement avec une requête GET.
Le fichier se nomme update.php.

```
<?php
?>
```

Le code entre les balises php vérifiera l'ID de contact, l'ID sera un paramètre dans l'URL, par exemple, http://localhost/phpcrud/update.php?id=1 obtiendra le contact avec l'ID de 1, puis nous pouvons gérer la requête avec la méthode GET et exécuter une requête MySQL qui obtiendra le contact par ID.

<?=template_header('Read')?>
<?=template_footer()?>

Entre les balises template le modèle de la page de mise à jour, les valeurs d'entrée sont déjà spécifiées avec les colonnes de contact, la requête MySQL que nous avons créée précédemment obtiendra ces valeurs.

La page de suppression

La page de suppression sera utilisée pour supprimer des enregistrements du tableau Contacts . Avant qu'un utilisateur puisse supprimer un enregistrement, il devra le confirmer, cela empêchera une suppression accidentelle.

<?php
?>

Pour supprimer un enregistrement, le code vérifiera si la variable de requête GET " id " existe, si c'est le cas, vérifiez si l'enregistrement existe dans la table Contacts et confirmez à l'utilisateur s'il souhaite supprimer le contact ou non, une simple requête GET déterminera sur quel bouton l'utilisateur a cliqué (Oui ou Non).

<?=template_header('Delete')?>

<?=template_footer()?>

Le code entre ses balises est le modèle de la page de suppression, cela inclut les boutons Oui et Non (confirmation de suppression) et le message de sortie. Les boutons Oui et Non créeront une nouvelle requête GET qui confirmera le choix de l'utilisateur.