

TD : Introduction au développement web (Partie 2)

Dans cette deuxième partie, vous allez vous servir des acquis de la première partie mais nous allons intégralement changer les fonctionnalités et le but de l'application.

Créer une application de gestion de prompts IA

Contexte

Vous avez déjà développé une application web simple permettant de saisir des contacts, d'enregistrer les données dans une base, et de les afficher sous forme de liste.

C'est une excellente base pour comprendre le **fonctionnement d'une architecture web classique (HTML + PHP + MySQL)**.

Dans cette seconde partie, vous allez transformer cette application :

Créer un gestionnaire de prompts IA, comme ceux utilisés avec ChatGPT, MidJourney ou d'autres outils d'intelligence artificielle.

Objectif pédagogique

- Appliquer les mêmes mécanismes que précédemment (formulaire, insertion en base, affichage),
- mais dans un **contexte métier plus riche** et structuré.

L'application permettra :

- de **créer des prompts**,
- de les **classer par type** (`texte`, `image`, `code`, etc.),
- de les **lier à un modèle d'IA** (`GPT-4`, `CLaude`, MidJourney, etc.),
- et de filtrer ou d'afficher les prompts par catégorie.

Compétences visées

- Utiliser plusieurs **tables relationnelles** et comprendre les **clés étrangères**
- Réaliser des **formulaires avec listes déroulantes** (données dynamiques)
- Concevoir une base **plus modulaire et évolutive**
- Réaliser des requêtes SQL avec **jointures simples**
- Déployer une application plus proche d'un **cas réel d'utilisation**

Exemple d'usage attendu

"Je veux retrouver tous les prompts text-to-image utilisables avec MidJourney v5, dans la catégorie création visuelle."

"Je veux ajouter un nouveau prompt text-to-text sans l'attacher à un modèle particulier."

Etapes prévues

1. Concevoir la base de données avec plusieurs tables (Merise)
2. Créer les fichiers de configuration et les tables en SQL
3. Adapter le formulaire HTML
4. Insérer et afficher les données avec PHP
5. Bonus : ajouter des filtres de recherche ou d'export

Workflow

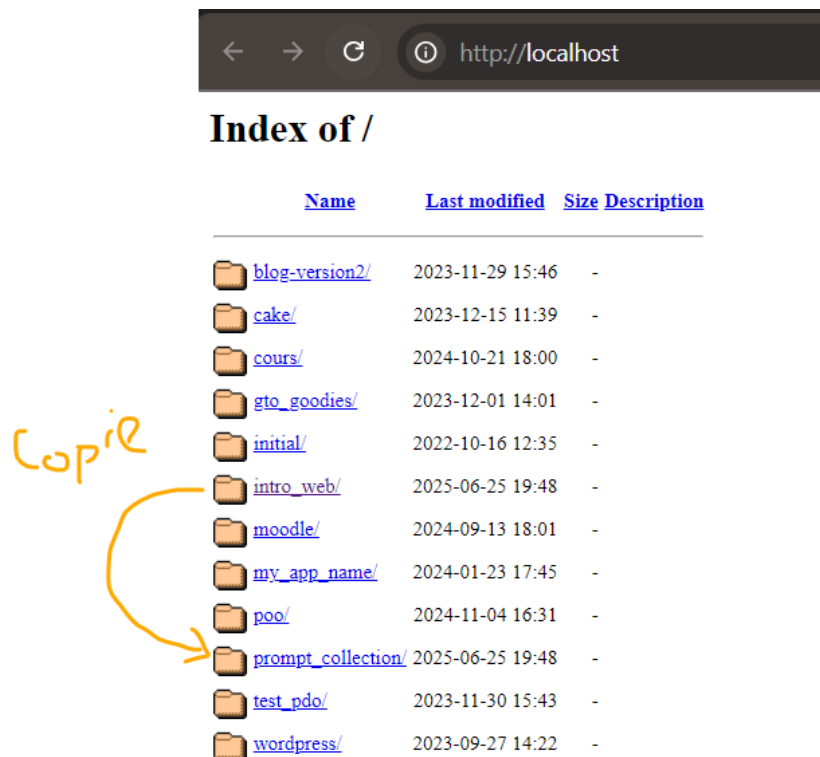
A ce stade vous avez deux choix :

- Refaire un nouveau projet à partir de zéro.
- Modifier le projet existant.

Dans les deux cas je vous conseille de travailler uniquement en local pour l'instant . Vous pourrez ensuite créer une bdd distante et uploader vos fichiers comme vous l'avez fait dans la première partie.

Duplication de l'ancien projet

👉 Dans le dossier `www`, dupliquez et renommez le projet `intro_web` en `prompt_collection` ou un autre nom



Partie 1 - mise en place de la base de données

Fonctionnalités de l'application (vue utilisateur)

Ajout d'un prompt

- Formulaire de création avec les champs suivants :
 - **Titre** du prompt (obligatoire)
 - **Contenu** du prompt (obligatoire)
 - **Type** (obligatoire, menu déroulant avec valeur par défaut : *Text-To-Text*)
 - **Outil** (facultatif, menu déroulant)
 - **Observation** (facultatif, zone de texte libre)
 - **Favori** (case à cocher pour marquer le prompt comme favori)
- Validation des champs obligatoires
- Insertion du prompt dans la base de données

1. Affichage des prompts enregistrés

- Liste ou tableau récapitulatif
- Pour chaque prompt : titre, type, outil (si présent), date, observation, statut favori
- Option possible : **filtrer par favoris, trier par date**

2. Modification ou suppression de prompts (fonctionnalité bonus ou facultative selon le temps)

- Modifier le contenu ou basculer un prompt en favori/non favori
- Supprimer un prompt de la base

3. Structure de navigation claire

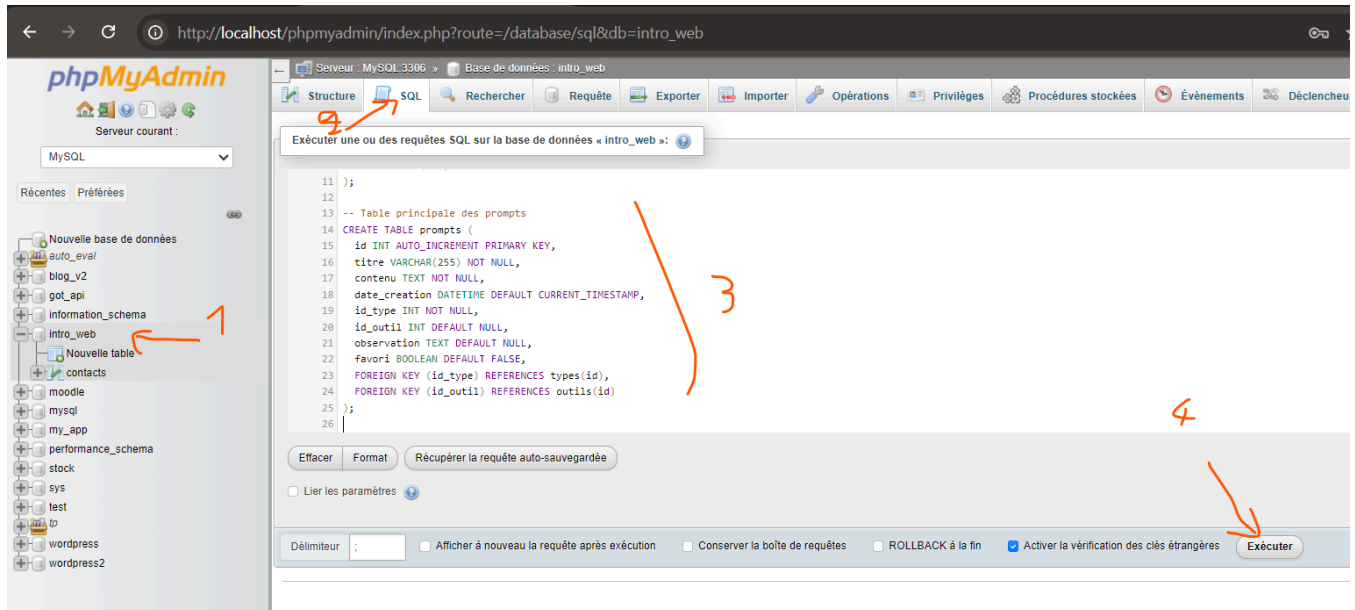
- Page d'accueil avec le formulaire d'ajout
- Lien vers la page de consultation des prompts
- Retour possible à la page d'accueil depuis la liste

Modification de la base de donnée

vous avez le choix de créer une nouvelle base de donnée, soit modifier la base utilisée dans la première partie. Bien que cela ne soit pas une bonne pratique, cela ne posera pas de problème puisque les nom de tables sont différents.

Dans PHPMyAdmin(local) onglet SQL

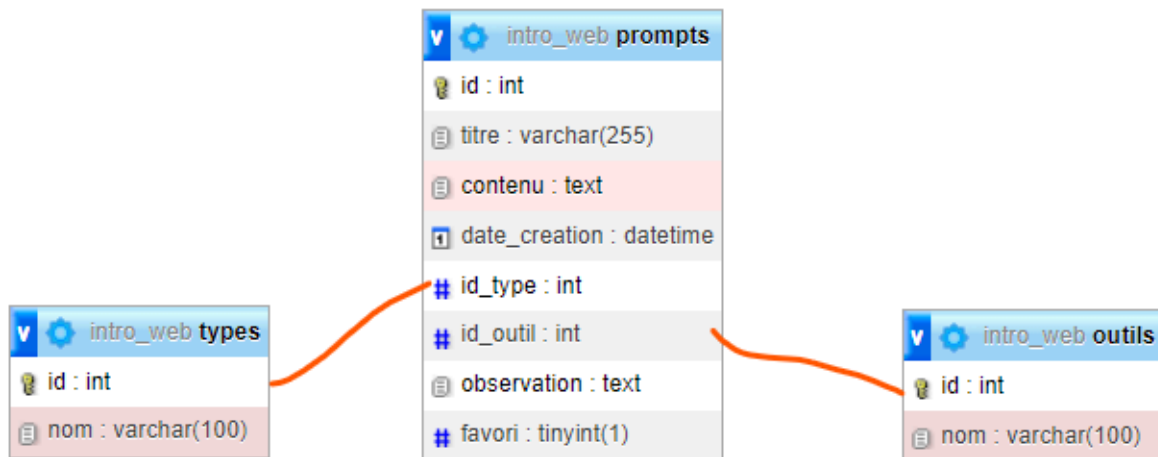
Insérer le script SQL suivant dans votre base de donnée locale



```
-- Table des types (obligatoire pour chaque prompt)
CREATE TABLE types (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nom VARCHAR(100) NOT NULL
);

-- Table des outils (facultatif pour les prompts)
CREATE TABLE outils (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nom VARCHAR(100) NOT NULL
);

-- Table principale des prompts
CREATE TABLE prompts (
  id INT AUTO_INCREMENT PRIMARY KEY,
  titre VARCHAR(255) NOT NULL,
  contenu TEXT NOT NULL,
  date_creation DATETIME DEFAULT CURRENT_TIMESTAMP,
  id_type INT NOT NULL,
  id_outil INT DEFAULT NULL,
  observation TEXT DEFAULT NULL,
  favori BOOLEAN DEFAULT FALSE,
  FOREIGN KEY (id_type) REFERENCES types(id),
  FOREIGN KEY (id_outil) REFERENCES outils(id)
);
```



Jeux d'essais

Un **jeu d'essai** (ou données de test) est un ensemble de **valeurs initiales** insérées dans une base de données pour permettre :

- de **tester rapidement** le bon fonctionnement des requêtes (ex. : est-ce que les prompts s'affichent bien ?)
- de **faciliter le développement** de l'application sans devoir entrer manuellement des données à chaque fois
- de **fournir des exemples** aux étudiants pour comprendre la logique de la base (types, outils, relations)
- d'éviter les erreurs liées à des tables vides (par exemple, un menu déroulant sans option si la table `types` est vide)

C'est une étape importante dans tout projet de développement, car elle permet de **valider la structure de la base** tout en aidant à construire l'interface de l'application.

Voici maintenant le **jeu d'essai SQL** à exécuter dans PHPMyAdmin après avoir créé la base :

```
-- Insertion des types de prompts
INSERT INTO types (nom) VALUES
    ('Text-To-Text'),
    ('Text-To-Image'),
    ('Text-To-Audio'),
    ('Image-To-Text');

-- Insertion des outils
INSERT INTO outils (nom) VALUES
    ('ChatGPT'),
    ('MidJourney'),
    ('DALL·E'),
    ('Bing Create');

-- Insertion de quelques prompts d'exemple
INSERT INTO prompts (titre, contenu, id_type, id_outil, observation, favori)
```

VALUES

```
('Résumé automatique', 'Fais un résumé en 5 lignes d'un texte de 500 mots.', 1, 1, 'Fonctionne bien avec GPT-3.5 et GPT-4', TRUE), ('Portrait surréaliste', 'Créer une image d'un chat en tenue d'astronaute sur Mars.', 2, 2, 'À ajuster selon la version de MidJourney', FALSE), ('Étiquette produit', 'Génère un slogan original pour un savon bio français.', 1, NULL, NULL, FALSE);
```

Ce jeu d'essai ajoute 4 types, 4 outils, et 3 prompts. Il permet d'avoir des valeurs visibles dès l'ouverture de la page de consultation, avec des cas variés (avec ou sans outil, avec ou sans observation, favori ou non).

Pour visualiser vous pouvez lancer la requête suivante :

```
SELECT * FROM prompts
```

	id	titre	contenu	date_creation	id_type	id_outil	observation	favori
<input type="checkbox"/> Éditer Copier Supprimer	1	Résumé automatique	Fais un résumé en 5 lignes d'un texte de 500 mots.	2025-06-27 11:23:49	1	1	Fonctionne bien avec GPT-3.5 et GPT-4	1
<input type="checkbox"/> Éditer Copier Supprimer	2	Portrait surréaliste	Créer une image d'un chat en tenue d'astronaute su...	2025-06-27 11:23:49	2	2	À ajuster selon la version de MidJourney	0
<input type="checkbox"/> Éditer Copier Supprimer	3	Étiquette produit	Génère un slogan original pour un savon bio frança...	2025-06-27 11:23:49	1	NULL NULL		0

Particularités

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	id	int			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
<input type="checkbox"/> 2	titre	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 3	contenu	text	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 4	date_creation	<u>datetime</u>			Oui	<u>CURRENT_TIMESTAMP</u>		DEFAULT_GENERATED	Modifier Supprimer Plus
<input type="checkbox"/> 5	id_type	int			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 6	id_outil	int			Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 7	observation	text	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 8	favori	<u>tinyint(1)</u>			Oui	0			Modifier Supprimer Plus

Type DATETIME DEFAULT CURRENT_TIMESTAMP

Le type DATETIME

- Le type `DATETIME` permet de stocker une date **et une heure** au format `YYYY-MM-DD HH:MM:SS`.
- Il est utile pour suivre des événements datés comme la création d'un enregistrement (`date_creation` dans notre cas).

La clause DEFAULT CURRENT_TIMESTAMP

- Elle permet de dire à MySQL :

« Si aucune date n'est précisée lors de l'insertion, utilise automatiquement la **date et l'heure actuelles** du serveur. »

- C'est pratique pour **journaliser automatiquement** les ajouts.

tinyint(1)

Dans **MySQL**, le type `BOOLEAN` (ou `BOOL`) **n'est qu'un alias de** `TINYINT(1)`.

- En SQL standard, un `BOOLEAN` est un type logique (valeurs possibles : `TRUE` ou `FALSE`).
- Mais **MySQL ne dispose pas d'un vrai type booléen** natif.
- Quand on écrit :

```
favori BOOLEAN DEFAULT FALSE
```

MySQL le **convertit automatiquement** en :

```
favori TINYINT(1) DEFAULT 0
```

Ce qui signifie :

- `0` → équivalent à `FALSE`
- `1` → équivalent à `TRUE`

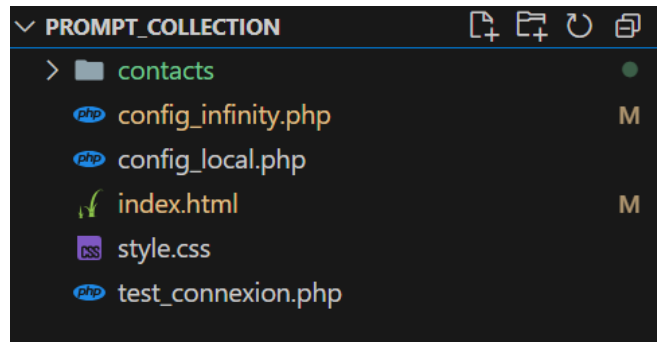
Conséquence dans phpMyAdmin

- Dans l'interface de phpMyAdmin, vous verrez le champ `favori` affiché comme `TINYINT(1)`.
- Mais vous pouvez toujours l'utiliser comme un booléen dans votre code PHP (test avec `if ($row['favori'])` fonctionne parfaitement).

Partie 2 - Ajout et lecture de prompts

👉 Dans le nouveau projet `prompt_collection` vous pouvez supprimer les fichiers `form.php` et `liste.php` ou les déplacer dans un sous-dossier en attendant.

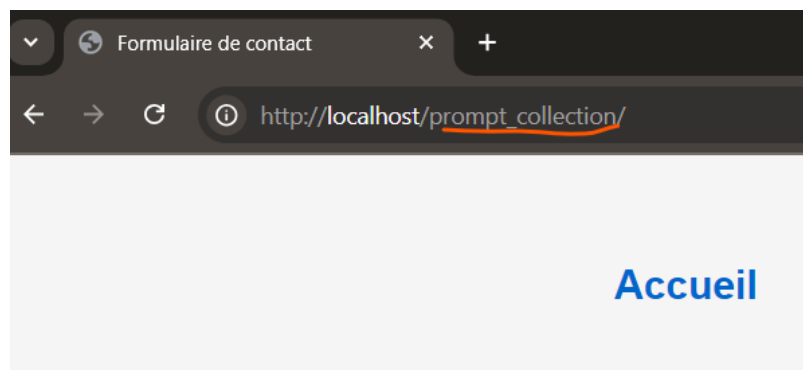
👉 Gardez les fichiers de config.



📄 Nous allons modifier le fichier `index.html`

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>Formulaire de contact</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Accueil</h1>
  </body>
</html>
```

👉 Vérifiez



📄 Créez un fichier `ajout_prompt.php` à la racine du site

```
<?php
// Connexion (à adapter selon environnement)
include 'config_local.php'; // ou config_infinity.php

// Récupération des types
$types = mysqli_query($conn, "SELECT id, nom FROM types");

// Récupération des outils
$outils = mysqli_query($conn, "SELECT id, nom FROM outils");
?>
```



```

<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  <title>Ajouter un prompt</title>
  <link rel="stylesheet" href="style.css">
</head>

<body>
  <h1>Ajouter un nouveau prompt</h1>

  <form action="traitement_prompt.php" method="POST">
    <label for="titre">Titre :</label><br>
    <input type="text" name="titre" required><br><br>

    <label for="contenu">Contenu :</label><br>
    <textarea name="contenu" rows="6" required></textarea><br><br>

    <label for="type">Type :</label><br>
    <select name="id_type" required>
      <?php while ($type = mysqli_fetch_assoc($types)): ?>
        <option value="<?= $type['id'] ?>"><?= htmlspecialchars($type['nom']) ?></option>
      <?php endwhile; ?>
    </select><br><br>

    <label for="outil">Outil (facultatif) :</label><br>
    <select name="id_outil">
      <option value="">-- Aucun --</option>
      <?php while ($outil = mysqli_fetch_assoc($outils)): ?>
        <option value="<?= $outil['id'] ?>"><?= htmlspecialchars($outil['nom']) ?></option>
      <?php endwhile; ?>
    </select><br><br>

    <label for="observation">Observation (facultatif) :</label><br>
    <textarea name="observation" rows="3"></textarea><br><br>

    <label>
      <input type="checkbox" name="favori" value="1"> Marquer comme favori
    </label><br><br>

    <button type="submit">Enregistrer le prompt</button>
  </form>
</body>

</html>

```

http://localhost/prompt_collection/ajout_prompt.php

Ajouter un nouveau prompt

Titre :

Contenu :

Type :

Text-To-Text ▼

Outil (facultatif) :

-- Aucun -- ▼

Observation (facultatif) :

☐ Marquer comme favori

Enregistrer le prompt

!? Pour l'instant n'essayez pas de valider, vous n'avez pas encore rempli le script

`traitement_prompt.php`

Explications

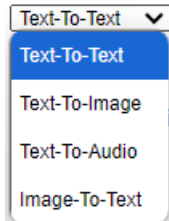
Particularités du fichier `ajout_prompt.php` qui **diffèrent** de ce qui a déjà été vu dans la première partie du TP :

Récupération dynamique des types et outils

```
$types = mysqli_query($conn, "SELECT id, nom FROM types");  
$outils = mysqli_query($conn, "SELECT id, nom FROM outils");
```

→ Ces deux requêtes permettent de **récupérer les listes à afficher dans les menus déroulants**. Cela rend l'application plus souple : on peut modifier les types ou outils directement dans la base, sans changer le code.

Type :



Boucles `while` pour remplir les `<select>`

```
<?php while ($type = mysqli_fetch_assoc($types)): ?>
<option value="<? $type['id'] ?>"><? htmlspecialchars($type['nom']) ?></option>
<?php endwhile; ?>
```

→ On utilise une **boucle PHP dans du HTML** pour générer dynamiquement les options du menu déroulant.

Le `<?= ... ?>` est une forme courte de `<?php echo ... ?>`.

Les deux-points `:` dans cette ligne :

```
<?php while ($type = mysqli_fetch_assoc($types)): ?>
```

sont une **alternative syntaxique** en PHP appelée "**syntaxe de contrôle alternative**", utilisée principalement dans les blocs **HTML + PHP imbriqués** (comme les boucles ou conditions).

- `:` = début du bloc (remplace `{`)
- `endwhile;`, `endif;`, `endforeach;` = fin du bloc (remplace `}`)

C'est une **option purement syntaxique**, souvent utilisée dans des fichiers de templates `.php` très imbriqués en HTML.

Protection avec `htmlspecialchars()`

```
<?= htmlspecialchars($type['nom']) ?>
```

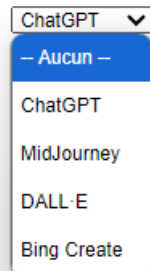
→ Sert à **éviter les injections HTML** ou les erreurs d'affichage si un nom contient des caractères spéciaux (`<`, `>`, `&`, etc.).

Option facultative dans le menu "outil"

```
<option value="">-- Aucun --</option>
```

→ Permet à l'utilisateur de ne **pas choisir d'outil**, ce qui est conforme à la règle métier.

Outil (facultatif) :



(facultatif) :

Case à cocher pour le champ booléen favori

```
<input type="checkbox" name="favori" value="1">
```

☐ Marquer comme favori

→ Si l'utilisateur coche la case, le champ favori vaudra 1.

S'il ne la coche pas, aucune valeur ne sera envoyée, donc il faudra gérer ça dans traitement_prompt.php.

Créez le script traitement_prompt.php à la racine du site

Le script traitement_prompt.php traite l'envoi du formulaire et insère un prompt dans la base. Il prend en compte les règles métiers :

- la sélection obligatoire du type
- l'outil facultatif
- le champ texte observation pouvant être NULL
- la case à cocher favori interprétée comme booléen

```
<?php
// Inclusion de la configuration en fonction de l'environnement
if (strpos($_SERVER['HTTP_HOST'], 'localhost') !== false) {
    include 'config_local.php';
} else {
    include 'config_infinity.php';
}

// Sécurisation et récupération des données du formulaire
$titre      = htmlspecialchars(trim($_POST['titre'] ?? ''));
$contenu    = htmlspecialchars(trim($_POST['contenu'] ?? ''));
$id_type    = intval($_POST['id_type'] ?? 0);
$id_outil   = isset($_POST['id_outil']) && $_POST['id_outil'] !== '' ?
    intval($_POST['id_outil']) : null;
```

```

$observation = !empty(trim($_POST['observation'])) ?
htmlspecialchars(trim($_POST['observation'])) : null;
$favori      = isset($_POST['favori']) ? 1 : 0;

// Vérification des champs obligatoires
if ($titre && $contenu && $id_type) {
    // Préparation de la requête SQL avec les bons champs
    $stmt = mysqli_prepare($conn,
        "INSERT INTO prompts (titre, contenu, id_type, id_outil, observation, favori)
        VALUES (?, ?, ?, ?, ?, ?)"
    );

    // Liaison des paramètres
    mysqli_stmt_bind_param($stmt, "ssisii",
        $titre,
        $contenu,
        $id_type,
        $id_outil,
        $observation,
        $favori
    );

    // Exécution
    if (mysqli_stmt_execute($stmt)) {
        echo "Le prompt a bien été enregistré.<br>";
        echo '<a href="ajout_prompt.php">Ajouter un autre prompt</a> | ';
        echo '<a href="liste_prompts.php">Voir les prompts</a>';
    } else {
        echo "Erreur à l'enregistrement : " . mysqli_stmt_error($stmt);
    }

    // Fermeture
    mysqli_stmt_close($stmt);
} else {
    echo "Veuillez remplir tous les champs obligatoires.";
}

mysqli_close($conn);

```

Explications

Ce code utilise `mysqli_prepare()` pour sécuriser les insertions (protection contre les injections SQL) et respecte les règles sur la présence ou non des champs facultatifs. Il est aussi compatible avec un champ `id_outil` pouvant être `NULL`.

(Voir Annexe pour en savoir un peu plus sur `mysqli_prepare()`)

⚠ Testez l'insertion en base de données.

http://localhost/prompt_collection/ajout_prompt.php

Ajouter un nouveau prompt

Titre :

Générer un entraînement personnalisé en vue de la préparation d'un triathlon

Contenu :

"En tant qu'expert en préparation physique et entraînement multisport, votre mission est de concevoir un programme d'entraînement personnalisé et complet pour un athlète de niveau intermédiaire se préparant à un triathlon dans les [INSÉRER NOMBRE] prochains mois.

Avant d'élaborer ce plan, vous devez poser cinq questions pertinentes et approfondies à l'athlète pour évaluer avec précision sa condition physique actuelle, ses objectifs spécifiques, ses contraintes de temps, ses préférences d'entraînement et tout antécédent médical ou blessure à prendre en compte. Sur la base de ces informations, vous créerez un calendrier d'entraînement détaillé, progressif et équilibré, couvrant les trois disciplines du triathlon (natation, cyclisme et course à pied), ainsi que

Type :

Text-To-Text ▼

Outil (facultatif) :

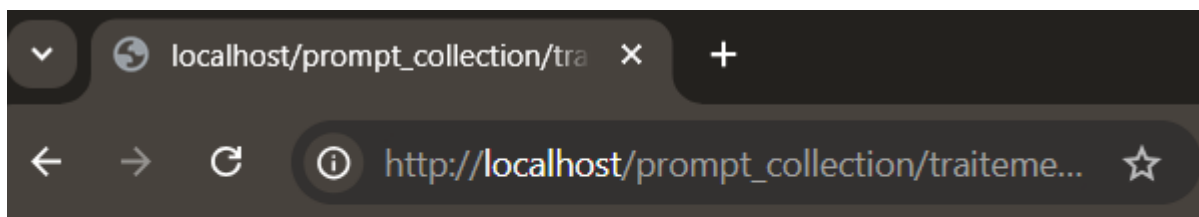
ChatGPT ▼

Observation (facultatif) :

PAs testé

☐ Marquer comme favori

Enregistrer le prompt



Le prompt a bien été enregistré.

[Ajouter un autre prompt](#) | [Voir les prompts](#)

Dans MySQL

		id	titre	contenu	date_creation	id_type	id_outil	observation	favori
<input type="checkbox"/>	Éditer Copier Supprimer	1	Résumé automatique	Fais un résumé en 5 lignes d'un texte de 500 mots.	2025-06-27 11:23:49	1	1	Fonctionne bien avec GPT-3.5 et GPT-4	1
<input type="checkbox"/>	Éditer Copier Supprimer	2	Portrait surréaliste	Crée une image d'un chat en tenue d'astronaute su...	2025-06-27 11:23:49	2	2	À ajuster selon la version de MidJourney	0
<input type="checkbox"/>	Éditer Copier Supprimer	3	Étiquette produit	Génère un slogan original pour un savon bio frança...	2025-06-27 11:23:49	1	NULL	NULL	0
<input type="checkbox"/>	Éditer Copier Supprimer	4	Générer un entraînement personnalis en vue de l...	"En tant qu'#039;expert en préparation physique e...	2025-06-27 12:42:31	1	1	Pas testé	0

📄 Créez le script `liste_prompts.php` à la racine du site

Le fichier `liste_prompts.php` permettra d'afficher tous les prompts enregistrés, triés du plus récent au plus ancien. Il affiche aussi le type, l'outil (si présent), les observations et si le prompt est marqué comme favori.

Fichier `liste_prompts.php`

```
<?php
// Détection de l'environnement (local ou InfinityFree)
if (strpos($_SERVER['HTTP_HOST'], 'localhost') !== false) {
    include 'config_local.php';
} else {
    include 'config_infinity.php';
}

// Requête SQL avec jointures pour récupérer type et outil (si présent)
$sql = "
    SELECT p.*, t.nom AS type_nom, o.nom AS outil_nom
    FROM prompts p
    JOIN types t ON p.id_type = t.id
    LEFT JOIN outils o ON p.id_outil = o.id
    ORDER BY p.date_creation DESC
";

$result = mysqli_query($conn, $sql);

// Début HTML
?>
<!DOCTYPE html>
<html lang="fr">
```

```

<head>
  <meta charset="UTF-8">
  <title>Liste des Prompts</title>
  <link rel="stylesheet" href="style.css">
</head>

<body>
  <h1>Liste des Prompts enregistrés</h1>

  <table>
    <thead>
      <tr>
        <th>Titre</th>
        <th>Contenu</th>
        <th>Type</th>
        <th>Outil</th>
        <th>Observation</th>
        <th>Favori</th>
        <th>Date</th>
      </tr>
    </thead>
    <tbody>
      <?php if (mysqli_num_rows($result) > 0): ?>
        <?php while ($row = mysqli_fetch_assoc($result)): ?>
          <tr>
            <td><?= $row['titre'] ?></td>
            <td><?= nl2br($row['contenu']) ?></td>
            <td><?= $row['type_nom'] ?></td>
            <td><?= $row['outil_nom'] ?? '-' ?></td>
            <td><?= $row['observation'] ?? '' ?></td>
            <td><?= $row['favori'] ? '✓' : '' ?></td>
            <td><?= $row['date_creation'] ?></td>
          </tr>
        <?php endwhile; ?>
      <?php else: ?>
        <tr>
          <td colspan="7">Aucun prompt enregistré.</td>
        </tr>
      <?php endif; ?>
    </tbody>
  </table>


  <p><a href="ajout_prompt.php">Ajouter un nouveau prompt</a></p>
</body>

</html>

<?php
mysqli_close($conn);
?>

```


Particularités importantes

- Utilisation de `JOIN` pour le type et `LEFT JOIN` pour l'outil (car il peut être `NULL`)
- Affichage d'un  si le prompt est marqué comme favori (`BOOLEAN`)
- Protection contre le HTML injecté avec `htmlspecialchars()`
- Affichage propre avec `nl2br()` pour les retours à la ligne dans le contenu

Testez la liste des prompts

http://localhost/prompt_collection/liste_prompts.php

Liste des Prompts enregistrés						
Titre	Contenu	Type	Outil	Observation	Favori	Date
Générer un entraînement personnalisé en vue de la préparation d'un triathlon	<p>"En tant qu'expert en préparation physique et entraînement multisport, votre mission est de concevoir un programme d'entraînement personnalisé et complet pour un athlète de niveau intermédiaire se préparant à un triathlon dans les [INSÉRER NOMBRE] prochains mois.</p> <p>Avant d'élaborer ce plan, vous devez poser cinq questions pertinentes et approfondies à l'athlète pour évaluer avec précision sa condition physique actuelle, ses objectifs spécifiques, ses contraintes de temps, ses préférences d'entraînement et tout antécédent médical ou blessure à prendre en compte. Sur la base de ces informations, vous créerez un calendrier d'entraînement détaillé, progressif et équilibré, couvrant les trois disciplines du triathlon (natation, cyclisme et course à pied), ainsi que le renforcement musculaire et la récupération. Ce programme devra inclure la fréquence, la durée et l'intensité des séances pour chaque discipline, des conseils nutritionnels adaptés, des stratégies de gestion de la fatigue et de prévention des blessures, ainsi que des repères de progression mesurables. Veuillez à adapter la charge d'entraînement au niveau intermédiaire de l'athlète, en proposant une progression graduelle et sûre vers</p>	Text-To-Text	ChatGPT	PAs testé		2025-06-27 12:42:31

Notre application commence à avoir pas mal de fonctionnalité mais son système de navigation est très pauvre. On va mettre en place un système de menu.

Pensez à commiter vos changements.

Partie 3 - Mise en place d'un Menu

Un **menu de navigation** va nous permettre de mieux s'orienter dans l'application et de passer facilement entre les pages : ajout de prompt, liste des prompts, accueil, etc.

On va créer un fichier séparé (`header.php`) et le **réutiliser** dans chaque page avec un `include`.

👉 Renommez le fichier `index.php` en `index.html` pour pouvoir insérer du PHP à l'intérieur.

Fichier `header.php`

```
<?php
$page = basename($_SERVER['PHP_SELF']);
echo '
<nav>
    <ul>
        <li><a href="index.php" ' . ($page === 'index.php' ? 'class="active"' : '') .
'>Accueil</a></li>
        <li><a href="ajout_prompt.php" ' . ($page === 'ajout_prompt.php' ? 'class="active"'
: '') . '>Ajouter un prompt</a></li>
        <li><a href="liste_prompts.php" ' . ($page === 'liste_prompts.php' ?
'class="active"' : '') . '>Liste des prompts</a></li>
    </ul>
</nav>';
```

Le principe ici est de **personnaliser automatiquement le lien actif** dans le menu en fonction de la page sur laquelle l'utilisateur se trouve. C'est une technique simple mais puissante pour **améliorer l'expérience utilisateur**, en donnant un repère visuel clair sur la navigation.

Voici comment cela fonctionne, étape par étape :

`$_SERVER['PHP_SELF']`

Cette superglobale contient le chemin du fichier actuellement exécuté, par exemple :

```
/mon_projet/liste_prompts.php
```

Une **superglobale** en PHP est une **variable spéciale toujours disponible**, quel que soit l'endroit du code où on se trouve (dans une fonction, une classe ou un fichier inclus). Ces variables sont prédéfinies par PHP et contiennent des **informations clés** sur l'environnement d'exécution, les requêtes du client, les formulaires, les fichiers envoyés, etc.

2. `basename($_SERVER['PHP_SELF'])`

Cette fonction extrait seulement le **nom du fichier** :

```
basename('/mon_projet/liste_prompts.php'); // donne "liste_prompts.php"
```

On stocke ce nom dans une variable `$page` pour pouvoir le comparer ensuite.

3. Ajout conditionnel de la classe `active`

Dans chaque lien du menu, on utilise une condition :

```
$page === 'index.php' ? 'class="active"' : ''
```

Cela signifie :

Si on est sur la page `index.php`, alors on ajoute `class="active"` au lien, sinon rien.

Exemple de rendu généré dynamiquement :

```
<li><a href="index.php" class="active">Accueil</a></li>
```

ou bien :

```
<li><a href="index.php">Accueil</a></li>
```

4. Utilisation dans le CSS

Il suffit ensuite de styliser la classe `active` dans votre `style.css` :

```
nav a.active {  
  font-weight: bold;  
  color: #0055aa;  
  text-decoration: underline;  
}
```

Avantage

- Code **réutilisable dans toutes les pages** sans le modifier.
- Ajoute une **surbrillance automatique** sur le lien actif sans JavaScript.
- Permet d'avoir une **navigation plus claire** pour l'utilisateur.

Ajoutez `include 'header.php';` dans chaque page PHP

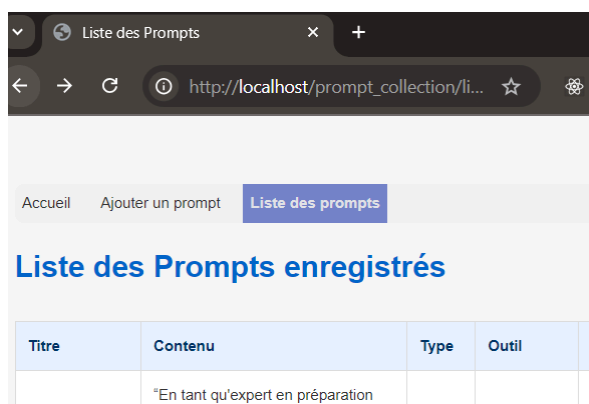
Usage cohérent dans toutes les pages

👉 Incluez `header.php` juste après la balise `<body>` dans chaque fichier `.php`. Cela rendra la navigation uniforme :

Par exemple, dans `liste_prompts.php` :

```
//.....
<body>
  <?php include 'header.php'; ?>
  <h1>Liste des Prompts enregistrés</h1>
//...
```

👉 Testez



Vous êtes de grands artistes, vous avez évidemment toute la latitude pour créer un template de menu plus joli que le miens.

👉 incluez le menu dans les autres pages

- `ajout_prompt.php`
- Accueil. c'est la page `index.html` renommez là en `index.php` pour pouvoir insérer du PHP

Ajoutez un peu de style dans `style.css`

- Gardez les sélecteurs CSS précédents
- Ajoutez à la fin du fichier

```
nav ul {
  list-style: none;
  padding: 0;
  display: flex;
  gap: 1rem;
  background: #f0f0f0;
  border-radius: 8px;
  margin-bottom: 1rem;
```

```
}

nav li {
  display: inline;
}

nav a {
  text-decoration: none;
  color: #333;
  padding: 0.5rem;
  display: block;
}

nav a:hover {
  background-color: #ddd;
  border-radius: 5px;
}

nav a.active {
  font-weight: bold;
  color: darkblue;
  text-decoration: underline;
}
```

Résultat

Un menu horizontal simple, lisible, réutilisable et facile à enrichir si vous ajoutez par la suite une page de recherche, de statistiques ou de favoris.

Récapitulatif du code

Vous trouverez le code sur https://github.com/trentindev/prompt_collection

Pour des raisons évidentes, le fichier config_infinity.php n'est pas intégré dans le dépôt. A vous de le créer avec vos identifiants en suivant la procédure du cours.

A vous de jouer...

La partie dirigée du TD arrête là...

A partir de maintenant c'est à vous de poursuivre.

Vous devez maintenant mettre en place les fonctionnalités suivantes.

Fonctionnalités obligatoires

La seule obligatoire est l'upload de vos fichiers sur infinityfree.fr pour que je puisse vérifier le site.

- **Uploadez votre projet sur infinityfree.fr**
- **Donnez moi le lien du site**

Fonctionnalités facultatives, classées par difficulté progressive

Choisissez ce que vous voulez ou créez en vous-même.

- **Modifier la page d'accueil**
Personnalisez `index.html` pour qu'elle donne envie de découvrir le site : nom, objectif, logo, explication courte.
- **Créer un `footer.php`**
Sur le modèle de `header.php`, ajoutez un pied de page réutilisable contenant des infos comme votre nom, une adresse email fictive ou une date de copyright.
- **Ajouter une page "À propos" ou "Crédits"**
Créez une nouvelle page (ex : `about.php`) pour expliquer le projet. Lien à insérer dans le menu.
- **Afficher un message de confirmation stylisé après l'ajout d'un prompt**
Ajoutez un petit encadré (`div` stylée avec `CSS`) qui s'affiche seulement après une insertion réussie.
- **Limiter la longueur du texte saisi dans les champs**
En HTML, utilisez les attributs `maxLength` sur les champs texte du formulaire.
- **Ajouter une colonne "Favori ★" dans la liste des prompts**
Affichez une étoile (ou autre symbole) quand le champ `favori` vaut `true`. Cela se gère avec une simple condition en PHP.
- **Trier les prompts favoris en haut de liste**
Ajoutez une clause `ORDER BY favori DESC, date_creation DESC` à votre requête dans `liste_prompts.php`.
- Ajouter un **page contacts** en réutilisant le code développé dans la partie une.
- **Ajouter une recherche par mot-clé**
Un champ `input` au-dessus de la liste permettrait de filtrer les prompts par mot dans leur contenu ou leur type.
- **Créer un système de suppression de prompts**
Ajouter un lien ou bouton dans chaque ligne de `liste_prompts.php` qui appelle un script `supprimer_prompt.php` avec confirmation.
- **Mettre en place un système de modification de prompt**
Permet d'éditer un prompt existant. Cela implique de pré-remplir un formulaire avec les données existantes (`edit_prompt.php`).
- **Créer un système de pagination**
Affichez les prompts par groupes de 5 ou 10 avec des boutons "suivant" / "précédent".
- Faites en sorte que votre application soit facilement consultable sur mobile (**Responsive**)

- **Ajouter une gestion rudimentaire d'utilisateur**

Créer un champ `auteur` (ou `email`) dans la base et permettre à chacun de ne voir que ses propres prompts.

Annexes

Pourquoi utiliser `mysqli_prepare()` ?

`mysqli_prepare()` est une fonction essentielle de l'extension `mysqli` en PHP. Elle permet de créer ce qu'on appelle une **requête préparée** (*prepared statement*), et son utilité est double : **sécurité** et **performance**.

Sécurité contre les injections SQL

Quand on insère directement des variables dans une requête SQL comme ceci :

```
$sql = "INSERT INTO prompts (titre) VALUES ('$titre')";
```

on court le risque que l'utilisateur injecte du code malveillant (comme `' OR 1=1;--`), ce qui pourrait compromettre la base.

Avec `mysqli_prepare()`, on sépare la structure de la requête et les données utilisateur :

```
$stmt = mysqli_prepare($conn, "INSERT INTO prompts (titre) VALUES (?)");  
mysqli_stmt_bind_param($stmt, "s", $titre);
```

→ Le paramètre `?` est **remplacé** par la valeur de `$titre`, sans jamais être interprété comme du SQL.

→ Le `"s"` dans la fonction `mysqli_stmt_bind_param()` indique le **type de données** de la variable que l'on va lier à la requête SQL préparée.

Cela signifie que la variable `$nom` est une **chaîne de caractères** (`string`). C'est le type le plus courant pour les données textuelles, comme un nom, un email, un message, etc.

Liste complète des types possibles :

Lettre	Type de données PHP	Signification
<code>s</code>	<code>string</code>	Chaîne de caractères
<code>i</code>	<code>integer</code>	Nombre entier
<code>d</code>	<code>double</code> (float)	Nombre à virgule flottante
<code>b</code>	<code>blob</code>	Données binaires (fichiers, etc.)

Optimisation des performances (surtout en boucle)

Si vous exécutez plusieurs fois une même requête avec des valeurs différentes, la requête préparée est **compilée une seule fois** côté serveur, ce qui accélère les traitements.

Comment ça fonctionne ?

Voici le cycle typique :

1. `mysqli_prepare($conn, $sql)` : prépare la requête **avec des ? à la place des valeurs**.
2. `mysqli_stmt_bind_param($stmt, $types, $val1, $val2, ...)` : lie les variables à insérer.
 - Le paramètre `$types` est une chaîne où :
 - `s` = string
 - `i` = integer
 - `d` = double (flottant)
 - `b` = blob
3. `mysqli_stmt_execute($stmt)` : exécute la requête préparée.

Exemples de failles évitées

Sans requête préparée :

```
$_POST['nom'] = "Robert'); DROP TABLE users; --";
```

→ Risque d'effacement de table entière si la requête est insérée directement.

Avec `mysqli_prepare()`, **la requête devient inoffensive**, car ce contenu est traité comme une chaîne de caractères, pas comme du code SQL.

Parfait, voici un **comparatif clair** à montrer à vos élèves, avec deux versions d'un même script :

- la première **vulnérable** à une injection SQL,
- la deuxième **sécurisée** avec `mysqli_prepare()`.

⚠ Exemple vulnérable (à ne jamais utiliser en production)

```
<?php
include 'config_local.php';

$nom = $_POST['nom']; // pas de nettoyage
$sql = "SELECT * FROM utilisateurs WHERE nom = '$nom'";

$result = mysqli_query($conn, $sql);

while ($row = mysqli_fetch_assoc($result)) {
    echo $row['nom'] . "<br>";
}
?>
```


Que se passe-t-il si un utilisateur envoie ceci dans un champ formulaire ?

```
' OR '1'='1
```

La requête devient alors :

```
SELECT * FROM utilisateurs WHERE nom = '' OR '1'='1'
```

→ Tous les utilisateurs de la base sont renvoyés !

→ Et on pourrait même aller plus loin : `'; DROP TABLE utilisateurs; --`

Les superglobales en PHP

Ce sont des **variables spéciales toujours disponibles**, quel que soit l'endroit du code où on se trouve (dans une fonction, une classe ou un fichier inclus).

Ces variables sont prédéfinies par PHP et contiennent des **informations clés** sur l'environnement d'exécution, les requêtes du client, les formulaires, les fichiers envoyés, etc.

Elles sont toutes représentées sous forme de **tableaux associatifs**.

Liste des superglobales les plus utilisées

Superglobale	Utilité principale
<code>\$_GET</code>	Contient les données envoyées via l'URL (requête GET)
<code>\$_POST</code>	Contient les données envoyées via un formulaire (requête POST)
<code>\$_REQUEST</code>	Combine <code>\$_GET</code> , <code>\$_POST</code> et <code>\$_COOKIE</code>
<code>\$_SERVER</code>	Donne des infos sur le serveur et le script en cours
<code>\$_SESSION</code>	Stocke des données de session côté serveur
<code>\$_COOKIE</code>	Contient les cookies envoyés par le client
<code>\$_FILES</code>	Gère les fichiers uploadés par formulaire
<code>\$_ENV</code>	Accède aux variables d'environnement
<code>\$_GLOBALS</code>	Accès global à toutes les variables définies dans l'espace global

Exemple simple : `$_GET`

```
// URL : page.php?nom=Laurent  
echo $_GET['nom']; // Affiche "Laurent"
```

Exemple avec `$_SERVER`

```
echo $_SERVER['PHP_SELF'];  
// Affiche le nom du fichier en cours, ex : "/projet/index.php"
```
