Cahier des Charges : Application CRUD WPF via API

Sommaire

1. Introduction

- **Titre du projet :** Application de Gestion CRUD "Patte à patte" (Admin/Backend)
- **Objectif :** Développer une application de bureau (desktop) en WPF C# permettant la gestion complète (Création, Lecture, Mise à jour, Suppression) des données essentielles du site web "Patte à patte" via des requêtes API sécurisées. Cette application est destinée à un usage interne pour l'administration du site.
- **Contexte :** Le site web "Patte à patte" nécessite un outil robuste pour la gestion de ses données de référence et de ses utilisateurs. Cette application WPF agira comme une interface d'administration, garantissant une manipulation des données efficace et sécurisée, indépendamment de l'interface publique du site.

2. Description Fonctionnelle Détaillée

2.1. Gestion des Entités

L'application doit permettre les opérations CRUD pour les entités suivantes :

1. **Gestion des Villes (CRUD)**
    - **Créer :** Ajouter une nouvelle ville (nom de la ville, code postal, etc.).
    - **Lire :** Afficher la liste de toutes les villes existantes avec leurs détails.
    - **Mettre à jour :** Modifier les informations d'une ville existante.
    - **Supprimer :** Retirer une ville de la base de données (avec confirmation et gestion des dépendances si une ville est liée à des utilisateurs).
2. **Gestion des Races (CRUD)**
    - **Créer :** Ajouter une nouvelle race de chien (nom de la race).
    - **Lire :** Afficher la liste de toutes les races existantes.
    - **Mettre à jour :** Modifier les informations d'une race existante.
    - **Supprimer :** Retirer une race.
3. **Gestion des Utilisateurs (CRUD)**
    - **Créer :** Ajouter un nouvel utilisateur (propriétaire ou dogsitter) avec ses informations de base (nom, prénom, email, rôle, etc.).
    - **Lire :** Afficher la liste de tous les utilisateurs.
    - **Mettre à jour :** Modifier les informations d'un utilisateur existant (changement de rôle).
    - **Supprimer :** Désactiver ou supprimer un compte utilisateur.

4. **Gestion des Types de Prestation (CRUD)**
    - **Créer :** Ajouter un nouveau type de prestation (ex: "Transport", "Visite").
    - **Lire :** Afficher la liste de tous les types de prestation.
    - **Mettre à jour :** Modifier le nom ou la description d'un type de prestation.
    - **Supprimer :** Retirer un type de prestation.

2.2. Authentification et Autorisation (pour l'application CRUD)

- **Connexion :** L'application WPF doit disposer d'un écran de connexion sécurisé pour les administrateurs.
- **Gestion des Sessions :** Maintenir une session utilisateur sécurisée après connexion.
- **Autorisation :** Les droits d'accès aux différentes fonctionnalités CRUD doivent être basés sur le rôle de l'utilisateur connecté (ex: seuls les administrateurs peuvent se connecter).

2.3. Interaction API

- Toutes les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer) doivent être effectuées via des appels à une API RESTful.
- L'application WPF doit envoyer et recevoir des données au format JSON.

3. Exigences Techniques

- **Plateformes Ciblées :**
    - **Client :** Application de bureau Windows (WPF C#).
    - **Backend API :** L'application s'appuiera sur une API existante ou à développer, idéalement basée sur un framework robuste (ex: Laravel) pour gérer les requêtes HTTP.
- **Langages et Frameworks :**
    - **Client :** C#, WPF (Windows Presentation Foundation).
    - **Communication :** Utilisation de HttpClient pour les requêtes API asynchrones.
- **Base de Données :** MySQL pour la persistance des données. L'application WPF n'interagit pas directement avec la base de données, mais via l'API.
- **Architecture API :**
    - API RESTful, utilisant les méthodes HTTP standard (GET, POST, PUT, DELETE).

    - Format de données : JSON pour les requêtes et les réponses.
    - Endpoints clairs et logiques pour chaque opération CRUD sur chaque entité.
- **Sécurité :**
    - **Authentification API :** Utilisation de jetons d'authentification (ex: JWT) pour sécuriser les appels API après la connexion de l'administrateur.
    - **Sécurité des données :** Toutes les communications entre l'application WPF et l'API doivent se faire via HTTPS.

3.1. Exigence (UX/UI)

- **Interface Intuitive :** L'interface utilisateur doit être claire, logique et facile à naviguer pour les administrateurs.
- **Design Consistant :** Utilisation de composants WPF standards et d'un style visuel cohérent.

4. Annexes

- **Documentation Technique :**
    - **Cas d'utilisation :** Scénarios détaillés des interactions utilisateur avec le système.
    - **Diagrammes d'architecture :** Vue d'ensemble des composants techniques et de leurs interactions.
- **Lien GitHub / Dépôt de Code :** Dépôt centralisé pour le code source du projet.

Ce cahier des charges est conçu pour fournir une vision claire et structurée du projet, assurant ainsi que toutes les parties prenantes partagent les mêmes attentes et objectifs. Il servira de référence tout au long du cycle de vie du développement.

-